Felix_prototype_V0.2

November 11, 2018

1 Felix prototype

Version 0.2
Date 10/11/2018
Model used : **Random Forest** Classifier on features selected using various techniques including **lasso**
Clustering method used : **Hierarchical clustering** using **ward metric** based on 6 **NOT variable**

```
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
        import itertools
        import pickle
        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import StandardScaler
        from sklearn.model_selection import cross_val_score, GridSearchCV
        from sklearn.ensemble import RandomForestClassifier
        from sklearn.metrics import confusion_matrix, f1_score, precision_score, recall_score
        from sklearn.model_selection import StratifiedKFold
        from sklearn.utils import resample
```

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

```
In [3]: # loading data
        file = path_data / Path("dataset.csv")
        with Path.open(file, 'rb') as fp:
            dataset = pd.read_csv(fp,  encoding='utf-8',low_memory=False, index_col = 0)
```

1.0.1 Features scope and selection strategy

Features are selected using lasso on the full scope of feature. The 50 more important features (logistic regression coef ranking) are kept regardless of their activability

```
In [4]: # load feature sets
        filename = path_dump / Path("dict_features_sets.sav")
        with open(filename, 'rb') as fp:
             dict_features_sets = pickle.load(fp)

In [5]: [k for k in dict_features_sets.keys()]

Out[5]: ['all_features',
         'cdv_features',
         'additional_features',
         'insee_features',
         'insee_demographics_features',
         'insee_recreation_features',
         'cat_features',
         'quant_features',
         'non_redundant_cdv_features',
         'cdv_actionable_individual_1_features',
         'cdv_actionable_individual_2_features',
         'cdv_actionable_individual_3_features',
         'cdv_actionable_individual_4_features',
         'cdv_actionable_admin_1_features',
         'cdv_actionable_admin_2_features',
         'cdv_actionable_admin_3_features',
         'cdv_actionable_admin_4_features',
         'cdv_actionable_admin_5_features',
         'insee_recreation_actionable_admin_1_features',
         'insee_recreation_actionable_admin_3_features',
         'insee_recreation_actionable_admin_4_features',
         'insee_demographics_actionable_admin_1_features',
         'insee_demographics_actionable_admin_2_features',
         'insee_demographics_actionable_admin_3_features',
         'insee_demographics_actionable_admin_4_features',
         'insee_demographics_actionable_admin_5_features',
         'insee_environment_score_features',
         'insee_recreation_score_features',
         'usual_common_scope_features',
         'usual_synthetic_scope_features',
         'RFE_LinearSVC_100_features',
         'RFE_LinearSVC_50_features',
         'RFE_LinearSVC_20_features',
         'RFE_LinearSVC_10_features',
         'RFE_RandomForestClassifier_100_features',
         'RFE_RandomForestClassifier_50_features',
         'RFE_RandomForestClassifier_20_features',
         'RFE_RandomForestClassifier_10_features',
         'RFE_LogisticRegression_100_features',
         'RFE_LogisticRegression_50_features',
         'RFE_LogisticRegression_20_features',
         'RFE_LogisticRegression_10_features',
         'SelectFromModel_LinearSCV_features',
         'SelectFromModel_LogisticRegression_features']
```

```
In [6]: usual_common_scope_features = dict_features_sets.get('usual_common_scope_features', set(
        # retreiving actionable features from disk
        cdv_actionable_individual_1_features = dict_features_sets.get('cdv_actionable_individual
        cdv_actionable_individual_2_features = dict_features_sets.get('cdv_actionable_individual
        cdv_actionable_admin_1_features = dict_features_sets.get('cdv_actionable_admin_1_feature
        cdv_actionable_admin_2_features = dict_features_sets.get('cdv_actionable_admin_2_feature
        insee_recreation_actionable_admin_1_features = dict_features_sets.get('insee_recreation_
        insee_recreation_actionable_admin_2_features = dict_features_sets.get('insee_recreation_
        insee_environment_actionable_admin_1_features = dict_features_sets.get('insee_environmen
        insee_environment_actionable_admin_2_features = dict_features_sets.get('insee_environmen
        insee_demographics_actionable_admin_1_features = dict_features_sets.get('insee_demograph
        insee_demographics_actionable_admin_2_features = dict_features_sets.get('insee_demograph
        # defining sets of actionable features
        actionable_individual_1_features = cdv_actionable_individual_1_features
        actionable_individual_2_features = cdv_actionable_individual_2_features
        actionable_admin_1_features = cdv_actionable_admin_1_features | insee_recreation_actiona
        actionable_admin_2_features = cdv_actionable_admin_2_features | insee_recreation_actiona

        RFE_LogisticRegression_10_features = dict_features_sets['RFE_LogisticRegression_10_featu
        RFE_LogisticRegression_20_features = dict_features_sets['RFE_LogisticRegression_20_featu
        RFE_LogisticRegression_50_features = dict_features_sets['RFE_LogisticRegression_50_featu
        RFE_LogisticRegression_100_features = dict_features_sets['RFE_LogisticRegression_100_fea

        RFE_RandomForestClassifier_100_features = dict_features_sets['RFE_RandomForestClassifier
        RFE_RandomForestClassifier_20_features = dict_features_sets['RFE_RandomForestClassifier_
        RFE_RandomForestClassifier_50_features = dict_features_sets['RFE_RandomForestClassifier_
        RFE_RandomForestClassifier_10_features = dict_features_sets['RFE_RandomForestClassifier_

        RFE_LinearSVC_100_features = dict_features_sets['RFE_LinearSVC_100_features']
        RFE_LinearSVC_50_features =  dict_features_sets['RFE_LinearSVC_50_features']
        RFE_LinearSVC_20_features =  dict_features_sets['RFE_LinearSVC_20_features']
        RFE_LinearSVC_10_features =  dict_features_sets['RFE_LinearSVC_10_features']
        SelectFromModel_LinearSCV_features = dict_features_sets['SelectFromModel_LinearSCV_featu
        SelectFromModel_LogisticRegression_features = dict_features_sets['SelectFromModel_Logist
```

```
In [7]: print(f"The {len(SelectFromModel_LogisticRegression_features)} features obtained using l
        print(f"{len(SelectFromModel_LogisticRegression_features & dict_features_sets.get('cdv_f
        print(list(SelectFromModel_LogisticRegression_features))
```

```
The 56 features obtained using lasso:
56 issues de l'étude CDV, 0 de l'insee, 0 calculées à partir des données insee
['INQROUT3_Non inquiet', 'RE_ALIM_Oui', 'INQALIM', 'TRANSFST_Oui', 'ETATSAN', 'SITUEMP3_Inactif'
```

1.0.2 Clustering method - feature used

Hierarchical clustering is used using 6 common "NOT_" variable

```
In [8]: # loading clustering
        file = path_data / Path("clustTest3.csv")
        with Path.open(file, 'rb') as fp:
            clustTest1 = pd.read_csv(fp,  encoding='utf-8',low_memory=False, sep=";", index_col
```

```
In [9]: cluster_name = ["","Civiques","Equilibrés","Flâneurs","Domestiques modérés","Solitaires"
```

1.1 Option 1 - Lasso only

```
In [10]: # choosing set of features
         scope = SelectFromModel_LogisticRegression_features
         print(f"number of features : {len(scope)} ow actionable")
         A = scope & dict_features_sets.get('cdv_actionable_admin_1_features',set())
         B = scope & dict_features_sets.get('cdv_actionable_individual_1_features',set())
         print(f"- at administrative level 1 : \t{len(A)}\n- at individual level 1 : \t{len(B)}"
```

```
number of features : 56 ow actionable
- at administrative level 1 :         27
- at individual level 1 :         25
```

1.1.1 Training set and test set preparation

```
In [11]: df = dataset.loc[:,:]
         # reducing problem to a 2 class classification problem
         df["HEUREUX_CLF"] = 0
         df.loc[df["HEUREUX"]==4, "HEUREUX_CLF"] = 1
         df.loc[df["HEUREUX"]==3, "HEUREUX_CLF"] = 1
         df.loc[df["HEUREUX"]==5, "HEUREUX_CLF"] = None

         scope = scope & set(dataset.columns)
         n_max = 2000

         df = df.loc[:,scope | {"HEUREUX_CLF"} ].dropna()
         features = df.loc[:,scope ].columns

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


         Xs, ys = resample(X, y, random_state=42)

         Xs = Xs.iloc[0:n_max,:]
         ys = ys.iloc[0:n_max]
         X_train, X_test, y_train, y_test = train_test_split(Xs, ys,
                                                              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: {y.shape[0]}\n- training set: \
         {y_train.shape[0]}\n- test set: {y_test.shape[0]}")
         print(f"Number of features: p={X_train.shape[1]}")
         print(f"Number of class: {len(np.unique(y))}")
         for c in np.unique(y):
             print(f"class {c:0.0f} : {100*np.sum(y==c)/len(y):0.1f}%")
```

```
Number exemple: 10788
- training set: 1600
- test set: 400
Number of features: p=56
Number of class: 2
class 0 : 35.0%
class 1 : 65.0%
```

1.1.2 Learning and model performance evaluation on full dataset (before clustering)

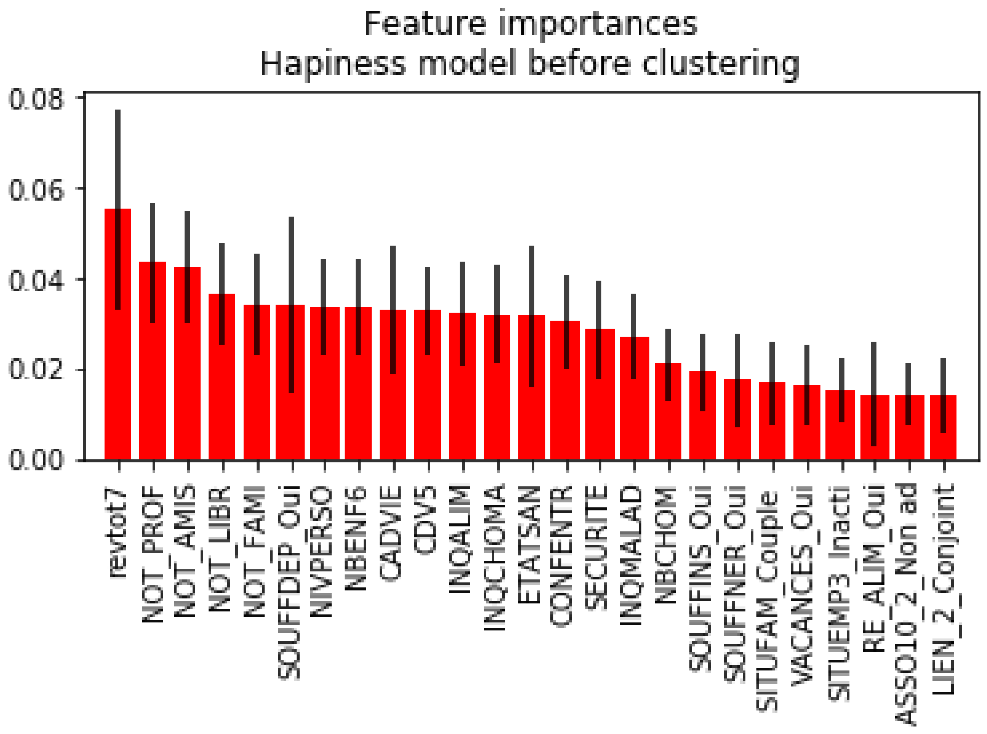
```
In [12]: startTime = time.time()
         n_estimators_range = [32,64,128,256,512]
         max_depth_range = [4,8,16,32,64]
         param_grid = dict(n_estimators=n_estimators_range, max_depth = max_depth_range)

         params = {'max_features' :'sqrt', 'random_state' : 32,
                   'min_samples_split' : 2, 'class_weight' : 'balanced'}
         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_} \n\
         Accuracy Score of cross valdation {100*grid.best_score_:0.2f}%")

         # 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)
         clf.fit(X_train, y_train)
         y_test_pred = clf.predict(X_test)

         print(f"Random Forest, p={X_train.shape[1]}")
         accuracy = clf.score(X_test, y_test)
         f1 = f1_score(y_test, y_test_pred)
         p = precision_score(y_test, y_test_pred)
         r = recall_score(y_test, y_test_pred)
         print(f"Model score\n- Accuracy : {accuracy*100:0.1f} %")
         print(f"- Precision : {p*100:0.1f} % (Happy # positive class)")
         print(f"- Recall : {r*100:0.1f} %")
         print(f"- F1 score : {f1*100:0.1f} %")
         res_full  = {
             'f1_score' : f1,
             'accuracy' : accuracy,
             'precision' : p,
             'recall' : r
         }
```

```
Determination of optimal hyperparameters in 43.2 s
Optimal values are {'max_depth': 16, 'n_estimators': 256}
Accuracy Score of cross valdation 75.56%
Random Forest, p=56
Model score
- Accuracy : 73.5 %
- Precision : 73.8 % (Happy # positive class)
- Recall : 90.1 %
- F1 score : 81.1 %
```

```
In [13]: importances = clf.feature_importances_
         std = np.std([tree.feature_importances_ for tree in clf.estimators_],
                      axis=0)
         indices = np.argsort(importances)[::-1]
         features_name = np.array(features)
         features_short_name_sorted = [ name[:15] for name in features_name[indices]]
         n_features_max = 25
         n_features = min(X.shape[1],n_features_max)
         print("Feature ranking:")

         # Plot the feature importances of the forest
         plt.figure()
         plt.gcf().subplots_adjust(bottom=0.4)
         plt.title("Feature importances\nHapiness model before clustering")
         plt.bar(range(n_features), importances[indices][:n_features],
                 color="r", yerr=std[indices[:n_features]], align="center")
         plt.xticks(range(n_features), features_short_name_sorted[:n_features], rotation=90)
         plt.xlim([-1, n_features])
```

```
        filename = path_fig / Path("feature_importance_option1.jpg")
        plt.savefig(filename, format='jpg')
        plt.show()

        for f in range(min(X.shape[1],n_features_max)):
            print("%d. feature %d -%s- (%f)" % (f + 1, indices[f],features_name[indices[f]], im
            if features_name[indices[f]] in actionable_individual_1_features:
                print("\tActionable at individual level (1)")
            if features_name[indices[f]] in actionable_individual_2_features:
                print("\tActionable at individual level (2)")
            if features_name[indices[f]] in actionable_admin_1_features:
                print("\tActionable at administrative level (1)")
            if features_name[indices[f]] in actionable_admin_2_features:
                print("\tActionable at administrative level (2)")
```

```
Feature ranking:
```



```
1. feature 27 -revtot7- (0.055216)
        Actionable at individual level (2)
        Actionable at administrative level (2)
2. feature 36 -NOT_PROF- (0.043599)
        Actionable at individual level (1)
        Actionable at administrative level (2)
```

```
3. feature 49 -NOT_AMIS- (0.042551)
        Actionable at individual level (1)
        Actionable at administrative level (2)
4. feature 34 -NOT_LIBR- (0.036881)
        Actionable at individual level (1)
        Actionable at administrative level (1)
5. feature 16 -NOT_FAMI- (0.034543)
        Actionable at individual level (1)
        Actionable at administrative level (2)
6. feature 48 -SOUFFDEP_Oui- (0.034192)
        Actionable at individual level (2)
        Actionable at administrative level (1)
7. feature 37 -NIVPERSO- (0.033697)
        Actionable at individual level (2)
        Actionable at administrative level (2)
8. feature 54 -NBENF6- (0.033558)
        Actionable at individual level (2)
        Actionable at administrative level (2)
9. feature 19 -CADVIE- (0.032957)
        Actionable at individual level (1)
        Actionable at administrative level (1)
10. feature 8 -CDV5- (0.032856)
        Actionable at individual level (2)
        Actionable at administrative level (1)
11. feature 2 -INQALIM- (0.032343)
        Actionable at individual level (1)
        Actionable at administrative level (1)
12. feature 26 -INQCHOMA- (0.032206)
        Actionable at individual level (1)
        Actionable at administrative level (1)
13. feature 4 -ETATSAN- (0.031754)
        Actionable at individual level (1)
        Actionable at administrative level (1)
14. feature 28 -CONFENTR- (0.030514)
        Actionable at individual level (1)
        Actionable at administrative level (1)
15. feature 21 -SECURITE- (0.028726)
        Actionable at individual level (2)
        Actionable at administrative level (1)
16. feature 44 -INQMALAD- (0.027184)
        Actionable at individual level (1)
        Actionable at administrative level (1)
17. feature 45 -NBCHOM- (0.021083)
        Actionable at individual level (1)
        Actionable at administrative level (1)
18. feature 15 -SOUFFINS_Oui- (0.019343)
        Actionable at individual level (2)
        Actionable at administrative level (1)
```

```
19. feature 53 -SOUFFNER_Oui- (0.017693)
        Actionable at individual level (2)
        Actionable at administrative level (1)
20. feature 42 -SITUFAM_Couple sans enfants- (0.016903)
        Actionable at individual level (2)
21. feature 35 -VACANCES_Oui- (0.016708)
        Actionable at individual level (1)
        Actionable at administrative level (1)
22. feature 5 -SITUEMP3_Inactif- (0.015247)
        Actionable at individual level (2)
        Actionable at administrative level (2)
23. feature 1 -RE_ALIM_Oui- (0.014443)
        Actionable at individual level (2)
        Actionable at administrative level (2)
24. feature 46 -ASSO10_2_Non adhérent- (0.014405)
        Actionable at individual level (1)
        Actionable at administrative level (1)
25. feature 10 -LIEN_2_Conjoint ou compagnon- (0.014060)
```

```
In [14]: print(f"number of features : {len(scope)} ow actionnable")
         A = scope & dict_features_sets.get('cdv_actionable_admin_1_features',set())
         B = scope & dict_features_sets.get('cdv_actionable_individual_1_features',set())
         print(f"- at administrative level 1 : \t{len(A)}\n- at individual level 1 : \t{len(B)}"
         important_features = set(features_name[indices][:10])
         C = A & important_features
         D = B & important_features
         print(f"- at administrative level 1 in top 10: \t{len(C)}\n- at individual level 1 in t
```

```
number of features : 56 ow actionnable
- at administrative level 1 :         27
- at individual level 1 :         25
- at administrative level 1 in top 10:         4
- at individual level 1 in top 10:         5
```

1.1.3 Learning and model performance evaluation on each clusters

```
In [15]: n_estimators_range = [16,32,64,128]
         max_depth_range = [2,4,8,16,32,64]
         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_weight' : 'balanced'
                  }
         #scope = ( SelectFromModel_LogisticRegression_features )  & set(dataset.columns)
         features = df.loc[:,scope].columns
```

```
In [16]: score_clustering_methods = []
         clustering_methods = clustTest1.columns[2:3]

         for method in clustering_methods:
             print("--------------------------------------------")
             print(f"\nAnalysis cluster method {method}")
             cluster_list = clustTest1[method].unique()
             print(f"liste of clusters : {cluster_list}")
             score_cluster = []
             for cluster in cluster_list:
                 index_scope = clustTest1.loc[clustTest1[method]==cluster,:].index
                 print(f"cluster {cluster} '{cluster_name[cluster]}' : {len(index_scope)} elemen

                 Xc = X.loc[index_scope.intersection(X.index),:]
                 yc = y[index_scope.intersection(X.index)]

                 Xs, ys = resample(Xc, yc, random_state=42)

                 Xs = Xs.iloc[0:n_max,:]
                 ys = ys.iloc[0:n_max]

                 X_train, X_test, y_train, y_test = train_test_split(Xs, ys,
                                                                     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: {ys.shape[0]}\n\
                 - training set: {y_train.shape[0]}\n\
                 - test set: {y_test.shape[0]}")
                 print(f"Number of features: p={X_train.shape[1]}")
                 print(f"Number of class: {len(np.unique(y))}")
                 for c in np.unique(y):
                     print(f"class {c:0.0f} : {100*np.sum(yc==c)/len(yc):0.1f}%")


                 startTime = time.time()
                 clf = RandomForestClassifier(**params)
                 grid = GridSearchCV(clf,
                                     scoring='accuracy',
                                     param_grid=param_grid)

                 grid.fit(X_train, y_train)
                 print(f"Optimal values are {grid.best_params_} \n\
         cross validation score {100*grid.best_score_:0.2f}%")
                 print()
```

```
                # Learning on full training set with optimals hyperparameters and score on test
                params_opt = {'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_opt).fit(X_train, y_train)


                y_test_pred = clf.predict(X_test)
                accuracy = clf.score(X_test, y_test)
                f1 = f1_score(y_test, y_test_pred)
                p = precision_score(y_test, y_test_pred)
                r = recall_score(y_test, y_test_pred)

                res  = {'f1_score' : f1,
                        'accuracy' : accuracy,
                        'precision' : p,
                        'recall' : r}


                importances = clf.feature_importances_
                std = np.std([tree.feature_importances_ for tree in clf.estimators_],
                             axis=0)
                indices = np.argsort(importances)[::-1]
                features_name = np.array(features)

                cl = {'cluster' : cluster,
                      'size' : len(index_scope),
                      'model' : 'RandomForestClassifier',
                      'params' : params_opt,
                      'metrics' : res,
                      'importances' : importances,
                      'sdt' : std,
                      'indices' : indices,
                      'features_name' : features_name
                     }

                score_cluster.append(cl)


            d = {'clustering_method' : method,
                 'cluster_scores' : score_cluster
                }
            score_clustering_methods.append(d)

----------------------------------------------
```

```
Analysis cluster method clust3
liste of clusters : [2 4 6 1 3 5]
cluster 2 'Equilibrés' : 3053 elements
Number exemple: 2000
        - training set: 1600
        - test set: 400
Number of features: p=56
Number of class: 2
class 0 : 34.7%
class 1 : 65.3%
Optimal values are {'max_depth': 16, 'n_estimators': 64}
cross validation score 80.06%

cluster 4 'Domestiques modérés' : 2359 elements
Number exemple: 2000
        - training set: 1600
        - test set: 400
Number of features: p=56
Number of class: 2
class 0 : 32.7%
class 1 : 67.3%
Optimal values are {'max_depth': 32, 'n_estimators': 128}
cross validation score 84.38%

cluster 6 'Domestiques stricts' : 2313 elements
Number exemple: 2000
        - training set: 1600
        - test set: 400
Number of features: p=56
Number of class: 2
class 0 : 32.8%
class 1 : 67.2%
Optimal values are {'max_depth': 16, 'n_estimators': 128}
cross validation score 83.56%

cluster 1 'Civiques' : 528 elements
Number exemple: 505
        - training set: 404
        - test set: 101
Number of features: p=56
Number of class: 2
class 0 : 48.3%
class 1 : 51.7%
Optimal values are {'max_depth': 8, 'n_estimators': 128}
cross validation score 81.93%

cluster 3 'Flâneurs' : 1384 elements
Number exemple: 1367
```

```
        - training set: 1093
        - test set: 274
Number of features: p=56
Number of class: 2
class 0 : 31.6%
class 1 : 68.4%
Optimal values are {'max_depth': 32, 'n_estimators': 128}
cross validation score 86.28%

cluster 5 'Solitaires' : 1494 elements
Number exemple: 1472
        - training set: 1177
        - test set: 295
Number of features: p=56
Number of class: 2
class 0 : 40.9%
class 1 : 59.1%
Optimal values are {'max_depth': 32, 'n_estimators': 128}
cross validation score 83.77%
```

1.1.4 Performance gain obtained using clustering

```
In [17]: # F1 score
         for score_method in score_clustering_methods:
             print(f"method {score_method['clustering_method']}:")
             average_score = 0
             total_size = 0
             for i, score_cluster in enumerate(score_method['cluster_scores']):
                 print(f"cluster {score_cluster['cluster']} '{cluster_name[score_cluster['cluste
                 average_score += score_cluster['metrics']['f1_score']*score_cluster['size']
                 total_size += score_cluster['size']

             average_score = average_score / total_size
             print(f"average f1 on clusters {100*average_score:0.1f}% gain {100*(average_score-r
```

```
method clust3:
cluster 2 'Equilibrés' : (3053), f1 macro 90.0%
cluster 4 'Domestiques modérés' : (2359), f1 macro 92.0%
cluster 6 'Domestiques stricts' : (2313), f1 macro 92.0%
cluster 1 'Civiques' : (528), f1 macro 89.6%
cluster 3 'Flâneurs' : (1384), f1 macro 92.1%
cluster 5 'Solitaires' : (1494), f1 macro 89.5%
average f1 on clusters 91.0% gain 9.9
```

```
In [18]: # accuracy
```

```
        for score_method in score_clustering_methods:
            print(f"method {score_method['clustering_method']}:")
            average_score = 0
            total_size = 0
            for i, score_cluster in enumerate(score_method['cluster_scores']):
                print(f"cluster {score_cluster['cluster']} '{cluster_name[score_cluster['cluste
                average_score = average_score + score_cluster['metrics']['accuracy']*score_clus
                total_size += score_cluster['size']
            average_score = average_score / total_size
            print(f"average accuracy on clusters {100*average_score:0.1f}% gain {100*(average_s
```

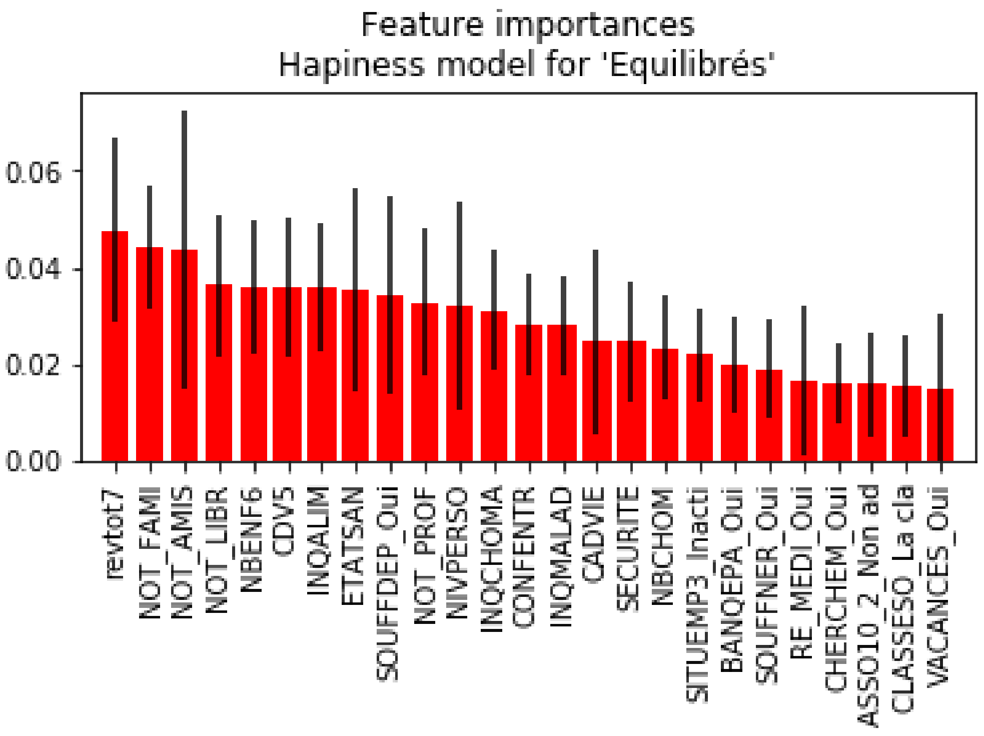
```
method clust3:
cluster 2 'Equilibrés' : (3053) accuracy 86.0%
cluster 4 'Domestiques modérés' : (2359) accuracy 88.2%
cluster 6 'Domestiques stricts' : (2313) accuracy 88.2%
cluster 1 'Civiques' : (528) accuracy 90.1%
cluster 3 'Flâneurs' : (1384) accuracy 88.3%
cluster 5 'Solitaires' : (1494) accuracy 87.5%
average accuracy on clusters 87.6% gain 14.1
```

1.1.5 Feature importance of the models & actionable variables

```
In [19]: # Feature importance by cluster
        for score_method in score_clustering_methods:
            print(f"method {score_method['clustering_method']}:")
            for i, score_cluster in enumerate(score_method['cluster_scores']):
                print(f"cluster {score_cluster['cluster']} ({score_cluster['size']}), f1 macro
                print(f"top 15 features:")
                indices = score_cluster['indices']
                features_name = score_cluster['features_name']
                importances = score_cluster['importances']
                features_short_name_sorted = [ name[:15] for name in features_name[indices]]

                # Plot the feature importances of the forest
                plt.figure()
                plt.gcf().subplots_adjust(bottom=0.4)
                plt.title(f"Feature importances\nHapiness model for '{cluster_name[score_cluste
                plt.bar(range(n_features), importances[indices][:n_features],
                        color="r", yerr=std[indices][:n_features], align="center")
                plt.xticks(range(n_features), features_short_name_sorted[:n_features], rotation
                plt.xlim([-1, n_features])
                filename = path_fig / Path(f"feature_importance_cluster_{score_cluster['cluster
                plt.savefig(filename, format='jpg')
                plt.show()

                for f in range(n_features_max):
```

```
            print("%d. feature %d -%s- (%f)" % (f + 1, indices[f],features_name[indices
            if features_name[indices[f]] in actionable_individual_1_features:
                print("\tActionable at individual level (1)")
            if features_name[indices[f]] in actionable_individual_2_features:
                print("\tActionable at individual level (2)")
            if features_name[indices[f]] in actionable_admin_1_features:
                print("\tActionable at administrative level (1)")
            if features_name[indices[f]] in actionable_admin_2_features:
                print("\tActionable at administrative level (2)")
```

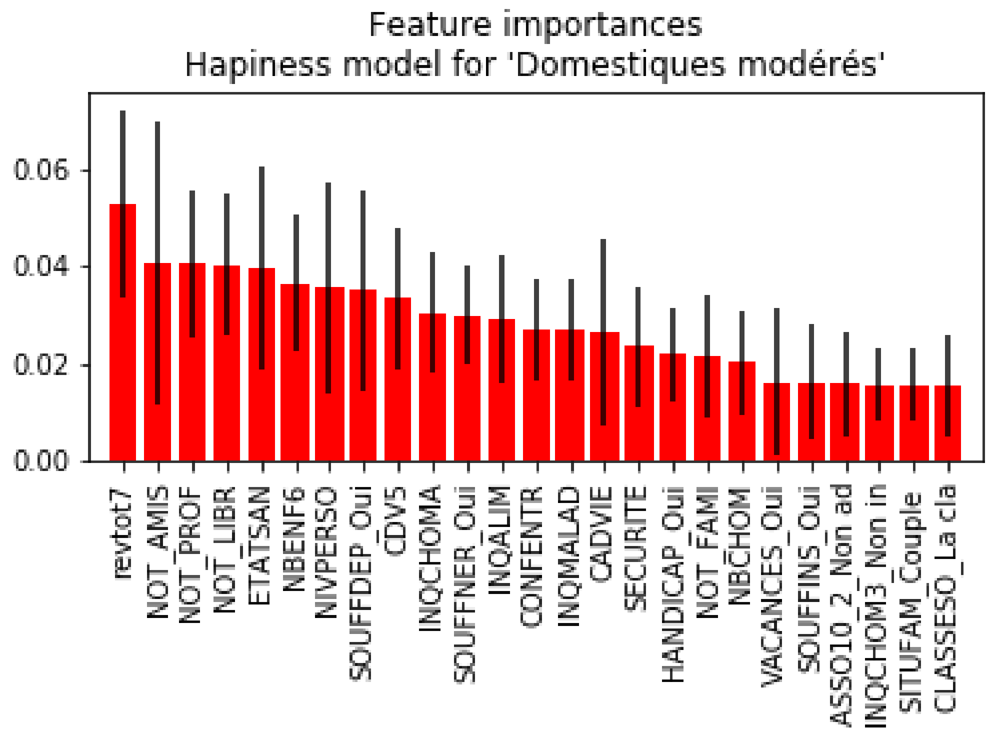
```
method clust3:
cluster 2 (3053), f1 macro 90.0%
top 15 features:
```



```
1. feature 27 -revtot7- (0.047657)
	Actionable at individual level (2)
	Actionable at administrative level (2)
2. feature 16 -NOT_FAMI- (0.044059)
	Actionable at individual level (1)
	Actionable at administrative level (2)
3. feature 49 -NOT_AMIS- (0.043665)
	Actionable at individual level (1)
	Actionable at administrative level (2)
```

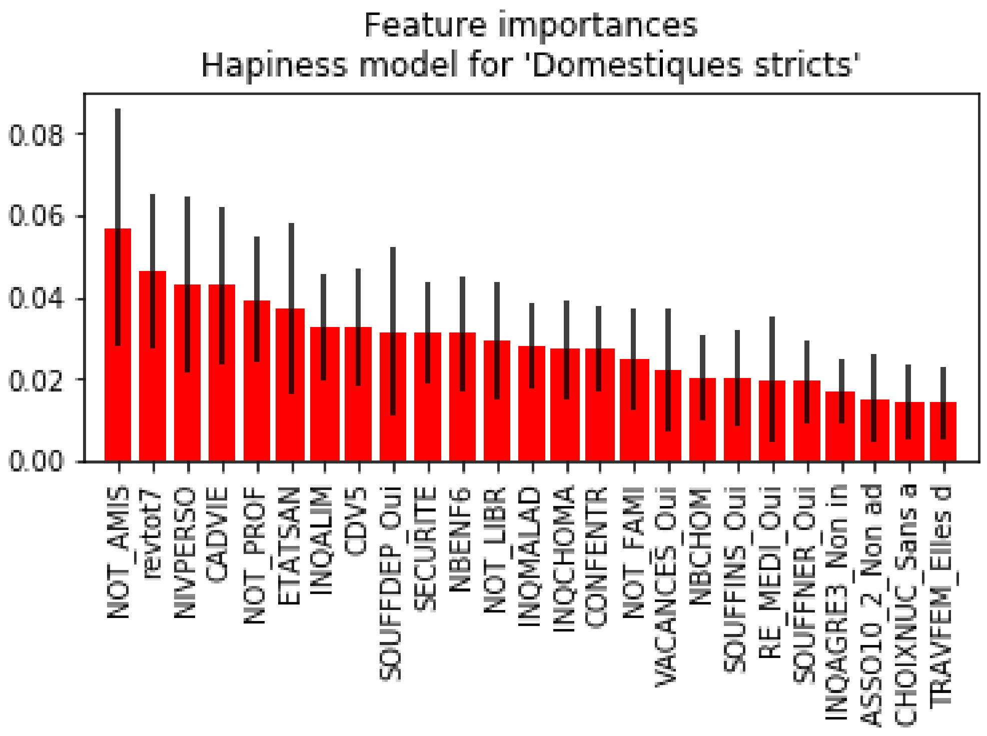
```
4. feature 34 -NOT_LIBR- (0.036328)
        Actionable at individual level (1)
        Actionable at administrative level (1)
5. feature 54 -NBENF6- (0.035803)
        Actionable at individual level (2)
        Actionable at administrative level (2)
6. feature 8 -CDV5- (0.035799)
        Actionable at individual level (2)
        Actionable at administrative level (1)
7. feature 2 -INQALIM- (0.035799)
        Actionable at individual level (1)
        Actionable at administrative level (1)
8. feature 4 -ETATSAN- (0.035452)
        Actionable at individual level (1)
        Actionable at administrative level (1)
9. feature 48 -SOUFFDEP_Oui- (0.034260)
        Actionable at individual level (2)
        Actionable at administrative level (1)
10. feature 36 -NOT_PROF- (0.032705)
        Actionable at individual level (1)
        Actionable at administrative level (2)
11. feature 37 -NIVPERSO- (0.031991)
        Actionable at individual level (2)
        Actionable at administrative level (2)
12. feature 26 -INQCHOMA- (0.031153)
        Actionable at individual level (1)
        Actionable at administrative level (1)
13. feature 28 -CONFENTR- (0.028112)
        Actionable at individual level (1)
        Actionable at administrative level (1)
14. feature 44 -INQMALAD- (0.028024)
        Actionable at individual level (1)
        Actionable at administrative level (1)
15. feature 19 -CADVIE- (0.024752)
        Actionable at individual level (1)
        Actionable at administrative level (1)
16. feature 21 -SECURITE- (0.024632)
        Actionable at individual level (2)
        Actionable at administrative level (1)
17. feature 45 -NBCHOM- (0.023313)
        Actionable at individual level (1)
        Actionable at administrative level (1)
18. feature 5 -SITUEMP3_Inactif- (0.021885)
        Actionable at individual level (2)
        Actionable at administrative level (2)
19. feature 22 -BANQEPA_Oui- (0.020176)
        Actionable at individual level (1)
        Actionable at administrative level (1)
```

```
20. feature 53 -SOUFFNER_Oui- (0.018885)
        Actionable at individual level (2)
        Actionable at administrative level (1)
21. feature 40 -RE_MEDI_Oui- (0.016657)
        Actionable at individual level (2)
        Actionable at administrative level (2)
22. feature 31 -CHERCHEM_Oui- (0.016081)
        Actionable at individual level (1)
        Actionable at administrative level (2)
23. feature 46 -ASSO10_2_Non adhérent- (0.015903)
        Actionable at individual level (1)
        Actionable at administrative level (1)
24. feature 18 -CLASSESO_La classe moyenne supérieure- (0.015272)
        Actionable at individual level (1)
        Actionable at administrative level (1)
25. feature 35 -VACANCES_Oui- (0.015246)
        Actionable at individual level (1)
        Actionable at administrative level (1)
cluster 4 (2359), f1 macro 92.0%
top 15 features:
```



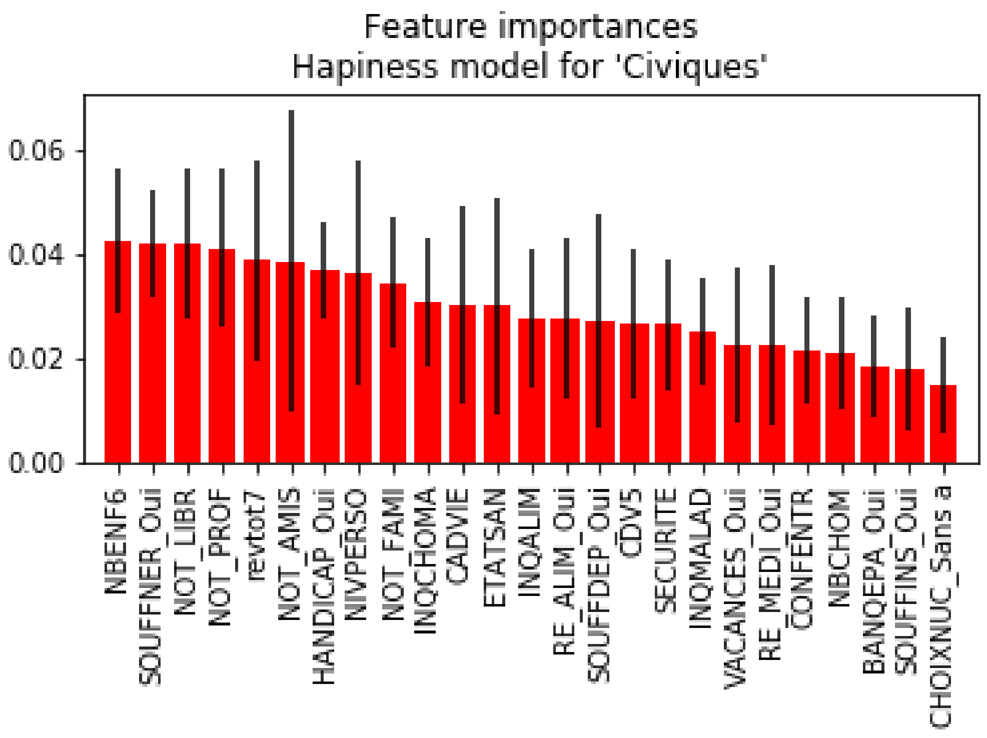
```
1. feature 27 -revtot7- (0.052969)
        Actionable at individual level (2)
        Actionable at administrative level (2)
2. feature 49 -NOT_AMIS- (0.040807)
        Actionable at individual level (1)
        Actionable at administrative level (2)
3. feature 36 -NOT_PROF- (0.040491)
        Actionable at individual level (1)
        Actionable at administrative level (2)
4. feature 34 -NOT_LIBR- (0.040371)
        Actionable at individual level (1)
        Actionable at administrative level (1)
5. feature 4 -ETATSAN- (0.039456)
        Actionable at individual level (1)
        Actionable at administrative level (1)
6. feature 54 -NBENF6- (0.036584)
        Actionable at individual level (2)
        Actionable at administrative level (2)
7. feature 37 -NIVPERSO- (0.035630)
        Actionable at individual level (2)
        Actionable at administrative level (2)
8. feature 48 -SOUFFDEP_Oui- (0.034990)
        Actionable at individual level (2)
        Actionable at administrative level (1)
9. feature 8 -CDV5- (0.033469)
        Actionable at individual level (2)
        Actionable at administrative level (1)
10. feature 26 -INQCHOMA- (0.030509)
        Actionable at individual level (1)
        Actionable at administrative level (1)
11. feature 53 -SOUFFNER_Oui- (0.029968)
        Actionable at individual level (2)
        Actionable at administrative level (1)
12. feature 2 -INQALIM- (0.029437)
        Actionable at individual level (1)
        Actionable at administrative level (1)
13. feature 28 -CONFENTR- (0.027193)
        Actionable at individual level (1)
        Actionable at administrative level (1)
14. feature 44 -INQMALAD- (0.027030)
        Actionable at individual level (1)
        Actionable at administrative level (1)
15. feature 19 -CADVIE- (0.026493)
        Actionable at individual level (1)
        Actionable at administrative level (1)
16. feature 21 -SECURITE- (0.023495)
        Actionable at individual level (2)
        Actionable at administrative level (1)
17. feature 41 -HANDICAP_Oui- (0.021881)
        Actionable at administrative level (2)
```

```
18. feature 16 -NOT_FAMI- (0.021367)
        Actionable at individual level (1)
        Actionable at administrative level (2)
19. feature 45 -NBCHOM- (0.020399)
        Actionable at individual level (1)
        Actionable at administrative level (1)
20. feature 35 -VACANCES_Oui- (0.016365)
        Actionable at individual level (1)
        Actionable at administrative level (1)
21. feature 15 -SOUFFINS_Oui- (0.016315)
        Actionable at individual level (2)
        Actionable at administrative level (1)
22. feature 46 -ASSO10_2_Non adhérent- (0.015830)
        Actionable at individual level (1)
        Actionable at administrative level (1)
23. feature 39 -INQCHOM3_Non inquiet- (0.015774)
        Actionable at individual level (1)
        Actionable at administrative level (1)
24. feature 42 -SITUFAM_Couple sans enfants- (0.015770)
        Actionable at individual level (2)
25. feature 18 -CLASSESO_La classe moyenne supérieure- (0.015742)
        Actionable at individual level (1)
        Actionable at administrative level (1)
cluster 6 (2313), f1 macro 92.0%
top 15 features:
```



```
1. feature 49 -NOT_AMIS- (0.057038)
        Actionable at individual level (1)
        Actionable at administrative level (2)
2. feature 27 -revtot7- (0.046253)
        Actionable at individual level (2)
        Actionable at administrative level (2)
3. feature 37 -NIVPERSO- (0.043320)
        Actionable at individual level (2)
        Actionable at administrative level (2)
4. feature 19 -CADVIE- (0.042762)
        Actionable at individual level (1)
        Actionable at administrative level (1)
5. feature 36 -NOT_PROF- (0.039357)
        Actionable at individual level (1)
        Actionable at administrative level (2)
6. feature 4 -ETATSAN- (0.037085)
        Actionable at individual level (1)
        Actionable at administrative level (1)
7. feature 2 -INQALIM- (0.032531)
        Actionable at individual level (1)
        Actionable at administrative level (1)
8. feature 8 -CDV5- (0.032395)
        Actionable at individual level (2)
        Actionable at administrative level (1)
9. feature 48 -SOUFFDEP_Oui- (0.031513)
        Actionable at individual level (2)
        Actionable at administrative level (1)
10. feature 21 -SECURITE- (0.031189)
        Actionable at individual level (2)
        Actionable at administrative level (1)
11. feature 54 -NBENF6- (0.031022)
        Actionable at individual level (2)
        Actionable at administrative level (2)
12. feature 34 -NOT_LIBR- (0.029325)
        Actionable at individual level (1)
        Actionable at administrative level (1)
13. feature 44 -INQMALAD- (0.027975)
        Actionable at individual level (1)
        Actionable at administrative level (1)
14. feature 26 -INQCHOMA- (0.027109)
        Actionable at individual level (1)
        Actionable at administrative level (1)
15. feature 28 -CONFENTR- (0.027101)
        Actionable at individual level (1)
        Actionable at administrative level (1)
```

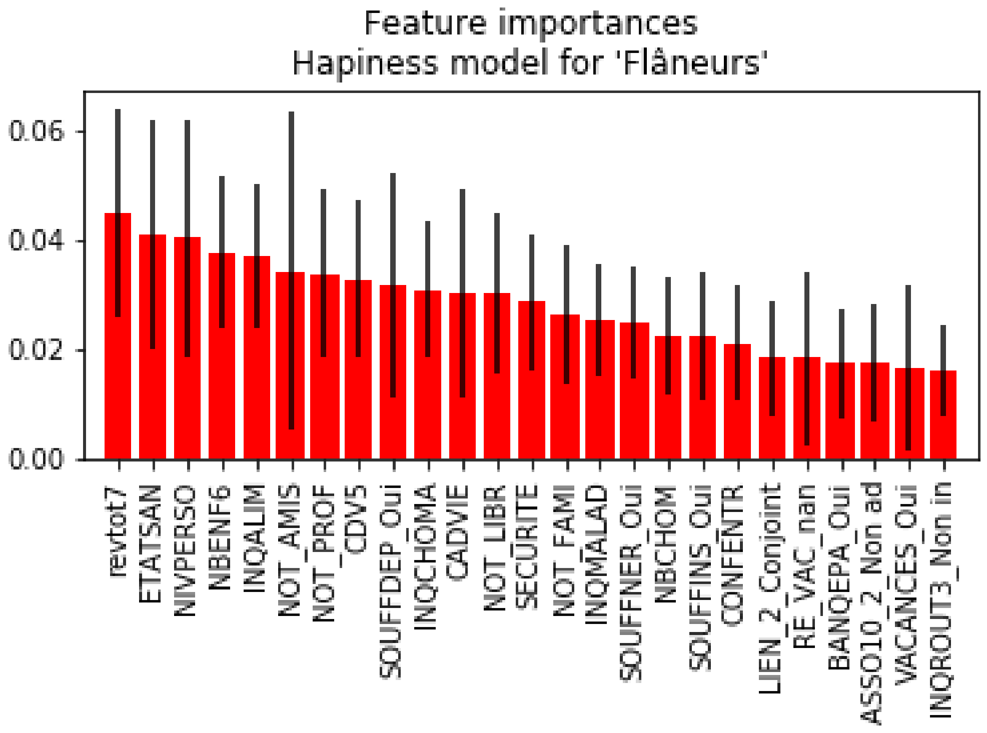
```
16. feature 16 -NOT_FAMI- (0.024628)
        Actionable at individual level (1)
        Actionable at administrative level (2)
17. feature 35 -VACANCES_Oui- (0.022356)
        Actionable at individual level (1)
        Actionable at administrative level (1)
18. feature 45 -NBCHOM- (0.020099)
        Actionable at individual level (1)
        Actionable at administrative level (1)
19. feature 15 -SOUFFINS_Oui- (0.020066)
        Actionable at individual level (2)
        Actionable at administrative level (1)
20. feature 40 -RE_MEDI_Oui- (0.019796)
        Actionable at individual level (2)
        Actionable at administrative level (2)
21. feature 53 -SOUFFNER_Oui- (0.019412)
        Actionable at individual level (2)
        Actionable at administrative level (1)
22. feature 11 -INQAGRE3_Non inquiet- (0.016937)
        Actionable at individual level (1)
        Actionable at administrative level (1)
23. feature 46 -ASSO10_2_Non adhérent- (0.015155)
        Actionable at individual level (1)
        Actionable at administrative level (1)
24. feature 17 -CHOIXNUC_Sans avis- (0.014534)
        Actionable at individual level (2)
        Actionable at administrative level (2)
25. feature 30 -TRAVFEM_Elles devraient travailler quand elles le désirent- (0.014191)
        Actionable at individual level (2)
        Actionable at administrative level (2)
cluster 1 (528), f1 macro 89.6%
top 15 features:
```



```
1. feature 54 -NBENF6- (0.042514)
        Actionable at individual level (2)
        Actionable at administrative level (2)
2. feature 53 -SOUFFNER_Oui- (0.042101)
        Actionable at individual level (2)
        Actionable at administrative level (1)
3. feature 34 -NOT_LIBR- (0.041892)
        Actionable at individual level (1)
        Actionable at administrative level (1)
4. feature 36 -NOT_PROF- (0.041140)
        Actionable at individual level (1)
        Actionable at administrative level (2)
5. feature 27 -revtot7- (0.038693)
        Actionable at individual level (2)
        Actionable at administrative level (2)
6. feature 49 -NOT_AMIS- (0.038585)
        Actionable at individual level (1)
        Actionable at administrative level (2)
7. feature 41 -HANDICAP_Oui- (0.036950)
        Actionable at administrative level (2)
8. feature 37 -NIVPERSO- (0.036182)
        Actionable at individual level (2)
        Actionable at administrative level (2)
```

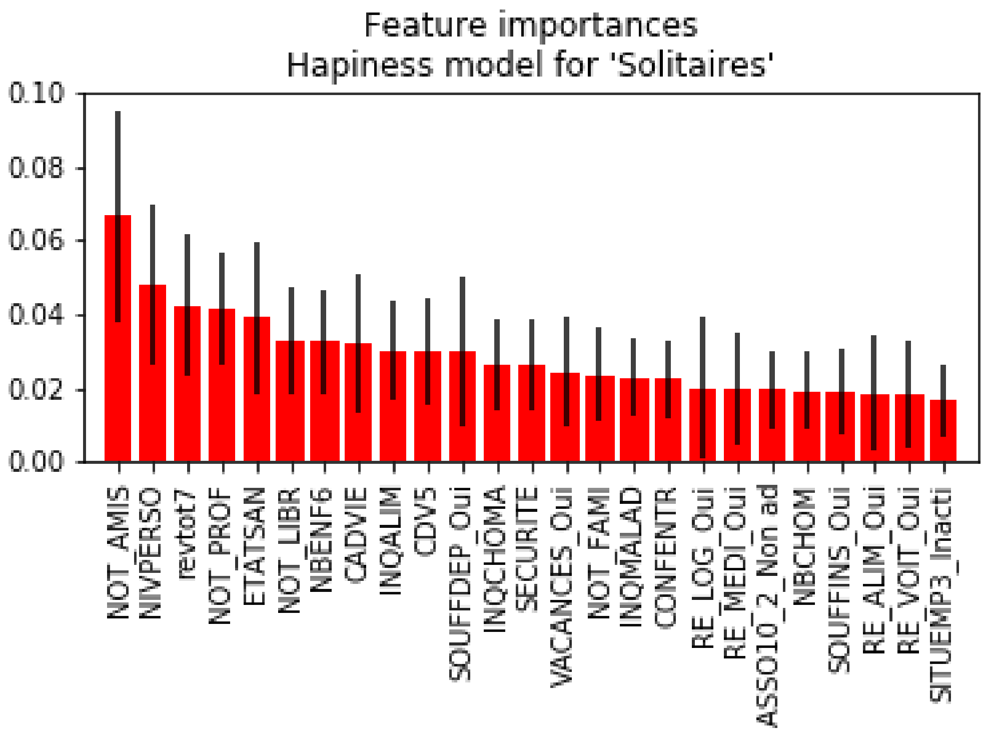
```
9. feature 16 -NOT_FAMI- (0.034522)
        Actionable at individual level (1)
        Actionable at administrative level (2)
10. feature 26 -INQCHOMA- (0.030862)
        Actionable at individual level (1)
        Actionable at administrative level (1)
11. feature 19 -CADVIE- (0.030240)
        Actionable at individual level (1)
        Actionable at administrative level (1)
12. feature 4 -ETATSAN- (0.030033)
        Actionable at individual level (1)
        Actionable at administrative level (1)
13. feature 2 -INQALIM- (0.027554)
        Actionable at individual level (1)
        Actionable at administrative level (1)
14. feature 1 -RE_ALIM_Oui- (0.027496)
        Actionable at individual level (2)
        Actionable at administrative level (2)
15. feature 48 -SOUFFDEP_Oui- (0.026946)
        Actionable at individual level (2)
        Actionable at administrative level (1)
16. feature 8 -CDV5- (0.026487)
        Actionable at individual level (2)
        Actionable at administrative level (1)
17. feature 21 -SECURITE- (0.026457)
        Actionable at individual level (2)
        Actionable at administrative level (1)
18. feature 44 -INQMALAD- (0.024914)
        Actionable at individual level (1)
        Actionable at administrative level (1)
19. feature 35 -VACANCES_Oui- (0.022484)
        Actionable at individual level (1)
        Actionable at administrative level (1)
20. feature 40 -RE_MEDI_Oui- (0.022319)
        Actionable at individual level (2)
        Actionable at administrative level (2)
21. feature 28 -CONFENTR- (0.021463)
        Actionable at individual level (1)
        Actionable at administrative level (1)
22. feature 45 -NBCHOM- (0.020811)
        Actionable at individual level (1)
        Actionable at administrative level (1)
23. feature 22 -BANQEPA_Oui- (0.018319)
        Actionable at individual level (1)
        Actionable at administrative level (1)
24. feature 15 -SOUFFINS_Oui- (0.017853)
        Actionable at individual level (2)
        Actionable at administrative level (1)
```

```
25. feature 17 -CHOIXNUC_Sans avis- (0.014681)
        Actionable at individual level (2)
        Actionable at administrative level (2)
cluster 3 (1384), f1 macro 92.1%
top 15 features:
```



```
1. feature 27 -revtot7- (0.044798)
        Actionable at individual level (2)
        Actionable at administrative level (2)
2. feature 4 -ETATSAN- (0.040740)
        Actionable at individual level (1)
        Actionable at administrative level (1)
3. feature 37 -NIVPERSO- (0.040127)
        Actionable at individual level (2)
        Actionable at administrative level (2)
4. feature 54 -NBENF6- (0.037606)
        Actionable at individual level (2)
        Actionable at administrative level (2)
5. feature 2 -INQALIM- (0.037001)
        Actionable at individual level (1)
        Actionable at administrative level (1)
6. feature 49 -NOT_AMIS- (0.034235)
        Actionable at individual level (1)
        Actionable at administrative level (2)
7. feature 36 -NOT_PROF- (0.033737)
        Actionable at individual level (1)
        Actionable at administrative level (2)
8. feature 8 -CDV5- (0.032764)
        Actionable at individual level (2)
        Actionable at administrative level (1)
9. feature 48 -SOUFFDEP_Oui- (0.031735)
        Actionable at individual level (2)
        Actionable at administrative level (1)
10. feature 26 -INQCHOMA- (0.030744)
        Actionable at individual level (1)
        Actionable at administrative level (1)
11. feature 19 -CADVIE- (0.030097)
        Actionable at individual level (1)
        Actionable at administrative level (1)
12. feature 34 -NOT_LIBR- (0.030080)
        Actionable at individual level (1)
        Actionable at administrative level (1)
13. feature 21 -SECURITE- (0.028604)
        Actionable at individual level (2)
        Actionable at administrative level (1)
14. feature 16 -NOT_FAMI- (0.026464)
        Actionable at individual level (1)
        Actionable at administrative level (2)
15. feature 44 -INQMALAD- (0.025425)
        Actionable at individual level (1)
        Actionable at administrative level (1)
16. feature 53 -SOUFFNER_Oui- (0.024982)
        Actionable at individual level (2)
        Actionable at administrative level (1)
17. feature 45 -NBCHOM- (0.022369)
        Actionable at individual level (1)
        Actionable at administrative level (1)
18. feature 15 -SOUFFINS_Oui- (0.022306)
        Actionable at individual level (2)
        Actionable at administrative level (1)
19. feature 28 -CONFENTR- (0.021033)
        Actionable at individual level (1)
        Actionable at administrative level (1)
20. feature 10 -LIEN_2_Conjoint ou compagnon- (0.018351)
21. feature 12 -RE_VAC_nan- (0.018308)
        Actionable at individual level (2)
        Actionable at administrative level (2)
22. feature 22 -BANQEPA_Oui- (0.017497)
        Actionable at individual level (1)
        Actionable at administrative level (1)
23. feature 46 -ASSO10_2_Non adhérent- (0.017463)
```

```
        Actionable at individual level (1)
        Actionable at administrative level (1)
24. feature 35 -VACANCES_Oui- (0.016800)
        Actionable at individual level (1)
        Actionable at administrative level (1)
25. feature 0 -INQROUT3_Non inquiet- (0.016194)
        Actionable at individual level (1)
        Actionable at administrative level (1)
cluster 5 (1494), f1 macro 89.5%
top 15 features:
```



```
1. feature 49 -NOT_AMIS- (0.066353)
        Actionable at individual level (1)
        Actionable at administrative level (2)
2. feature 37 -NIVPERSO- (0.047735)
        Actionable at individual level (2)
        Actionable at administrative level (2)
3. feature 27 -revtot7- (0.042365)
        Actionable at individual level (2)
        Actionable at administrative level (2)
4. feature 36 -NOT_PROF- (0.041131)
        Actionable at individual level (1)
        Actionable at administrative level (2)
```

```
5. feature 4 -ETATSAN- (0.038959)
        Actionable at individual level (1)
        Actionable at administrative level (1)
6. feature 34 -NOT_LIBR- (0.032890)
        Actionable at individual level (1)
        Actionable at administrative level (1)
7. feature 54 -NBENF6- (0.032387)
        Actionable at individual level (2)
        Actionable at administrative level (2)
8. feature 19 -CADVIE- (0.031988)
        Actionable at individual level (1)
        Actionable at administrative level (1)
9. feature 2 -INQALIM- (0.030055)
        Actionable at individual level (1)
        Actionable at administrative level (1)
10. feature 8 -CDV5- (0.029810)
        Actionable at individual level (2)
        Actionable at administrative level (1)
11. feature 48 -SOUFFDEP_Oui- (0.029698)
        Actionable at individual level (2)
        Actionable at administrative level (1)
12. feature 26 -INQCHOMA- (0.026401)
        Actionable at individual level (1)
        Actionable at administrative level (1)
13. feature 21 -SECURITE- (0.026207)
        Actionable at individual level (2)
        Actionable at administrative level (1)
14. feature 35 -VACANCES_Oui- (0.024314)
        Actionable at individual level (1)
        Actionable at administrative level (1)
15. feature 16 -NOT_FAMI- (0.023395)
        Actionable at individual level (1)
        Actionable at administrative level (2)
16. feature 44 -INQMALAD- (0.022752)
        Actionable at individual level (1)
        Actionable at administrative level (1)
17. feature 28 -CONFENTR- (0.022350)
        Actionable at individual level (1)
        Actionable at administrative level (1)
18. feature 29 -RE_LOG_Oui- (0.019795)
        Actionable at individual level (2)
        Actionable at administrative level (2)
19. feature 40 -RE_MEDI_Oui- (0.019517)
        Actionable at individual level (2)
        Actionable at administrative level (2)
20. feature 46 -ASSO10_2_Non adhérent- (0.019319)
        Actionable at individual level (1)
        Actionable at administrative level (1)
```

```
21. feature 45 -NBCHOM- (0.019135)
        Actionable at individual level (1)
        Actionable at administrative level (1)
22. feature 15 -SOUFFINS_Oui- (0.019126)
        Actionable at individual level (2)
        Actionable at administrative level (1)
23. feature 1 -RE_ALIM_Oui- (0.018479)
        Actionable at individual level (2)
        Actionable at administrative level (2)
24. feature 6 -RE_VOIT_Oui- (0.018183)
        Actionable at individual level (2)
        Actionable at administrative level (2)
25. feature 5 -SITUEMP3_Inactif- (0.016503)
        Actionable at individual level (2)
        Actionable at administrative level (2)
```

1.2 Option 2

```
In [20]: # choosing set of features
         scope = dict_features_sets.get('insee_environment_score_features', set())
         scope = scope | dict_features_sets.get('insee_recreation_score_features',set())
         #scope = scope | dict_features_sets.get('cdv_actionable_admin_1_features',set())
         scope = scope | RFE_LogisticRegression_20_features
         print(f"number of features : {len(scope)} ow actionable")
         A = scope & dict_features_sets.get('cdv_actionable_admin_1_features',set())
         B = scope & dict_features_sets.get('cdv_actionable_individual_1_features',set())
         print(f"- at administrative level 1 : \t{len(A)}\n- at individual level 1 : \t{len(B)}"
```

```
number of features : 34 ow actionable
- at administrative level 1 :         11
- at individual level 1 :         9
```

```
In [21]: df = dataset.loc[:,:]
         # reducing problem to a 2 class classification problem
         df["HEUREUX_CLF"] = 0
         df.loc[df["HEUREUX"]==4, "HEUREUX_CLF"] = 1
         df.loc[df["HEUREUX"]==3, "HEUREUX_CLF"] = 1
         df.loc[df["HEUREUX"]==5, "HEUREUX_CLF"] = None

         scope = scope & set(dataset.columns)
         n_max = 2000

         df = df.loc[:,scope | {"HEUREUX_CLF"} ].dropna()
         features = df.loc[:,scope ].columns

         X = df.loc[:,scope]
```

```
y = df["HEUREUX_CLF"]


Xs, ys = resample(X, y, random_state=42)

Xs = Xs.iloc[0:n_max,:]
ys = ys.iloc[0:n_max]

X_train, X_test, y_train, y_test = train_test_split(Xs, ys,
                                                     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: {y.shape[0]}\n- training set: \
{y_train.shape[0]}\n- test set: {y_test.shape[0]}")
print(f"Number of features: p={X_train.shape[1]}")
print(f"Number of class: {len(np.unique(y))}")
for c in np.unique(y):
    print(f"class {c:0.0f} : {100*np.sum(y==c)/len(y):0.1f}%")
```

```
Number exemple: 10858
- training set: 1600
- test set: 400
Number of features: p=34
Number of class: 2
class 0 : 34.9%
class 1 : 65.1%
```

```
In [22]: startTime = time.time()
         n_estimators_range = [32,64,128,256,512]
         max_depth_range = [4,8,16,32,64]
         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_weight' : 'balanced'}
         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_} \n\
         Accuracy Score of cross valdation {100*grid.best_score_:0.2f}%")
```

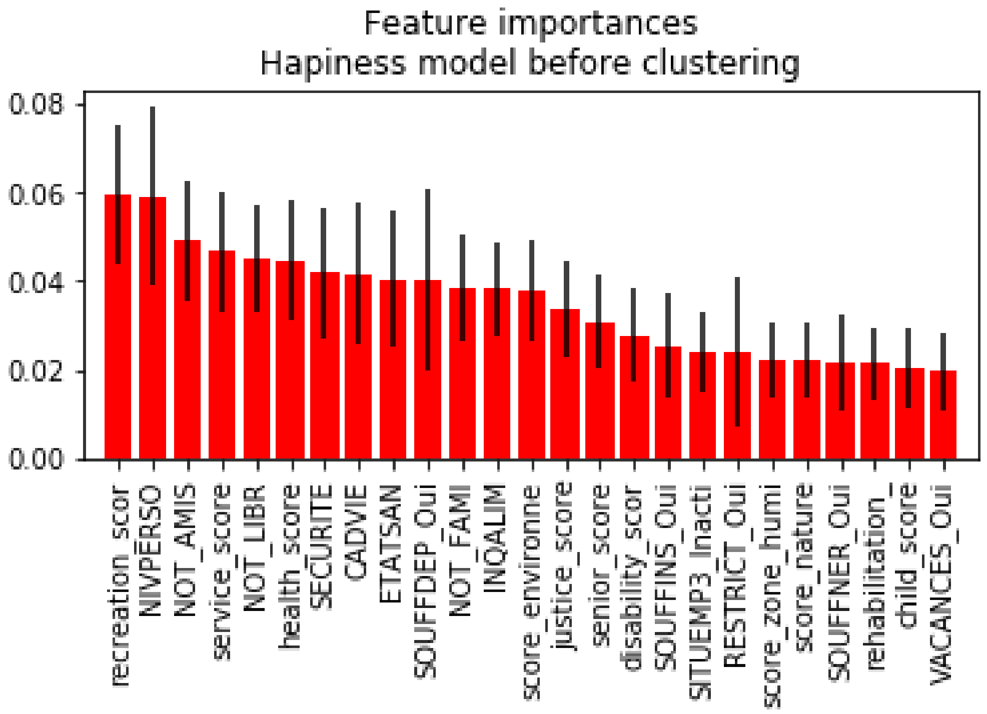
```
          # Learning on full training set with optimals hyperparameters and score on test set
          params = {'max_features' :'sqrt', 'random_state' : 32,
                    'min_samples_split' : 2, 'class_weight' : 'balanced',
                    'n_estimators' : grid.best_params_['n_estimators'],
                    'max_depth' : grid.best_params_['max_depth']}
          clf = RandomForestClassifier(**params).fit(X_train, y_train)
          clf.fit(X_train, y_train)
          y_test_pred = clf.predict(X_test)

          print(f"Random Forest, p={X_train.shape[1]}")
          accuracy = clf.score(X_test, y_test)
          f1 = f1_score(y_test, y_test_pred)
          p = precision_score(y_test, y_test_pred)
          r = recall_score(y_test, y_test_pred)
          print(f"Model score\n- Accuracy : {accuracy*100:0.1f} %")
          print(f"- Precision : {p*100:0.1f} % (Happy # positive class)")
          print(f"- Recall : {r*100:0.1f} %")
          print(f"- F1 score : {f1*100:0.1f} %")
          res_full  = {
              'f1_score' : f1,
              'accuracy' : accuracy,
              'precision' : p,
              'recall' : r
          }
```

```
Determination of optimal hyperparameters in 63.2 s
Optimal values are {'max_depth': 16, 'n_estimators': 256}
Accuracy Score of cross valdation 77.25%
Random Forest, p=34
Model score
- Accuracy : 73.5 %
- Precision : 71.9 % (Happy # positive class)
- Recall : 94.0 %
- F1 score : 81.5 %
```

```
In [23]: importances = clf.feature_importances_
         std = np.std([tree.feature_importances_ for tree in clf.estimators_],
                      axis=0)
         indices = np.argsort(importances)[::-1]
         features_name = np.array(features)
         features_short_name_sorted = [ name[:15] for name in features_name[indices]]
         print("Feature ranking:")

         n_features_max = 25
         n_features = min(X.shape[1],n_features_max)

         # Plot the feature importances of the forest
```

```
plt.figure()
plt.gcf().subplots_adjust(bottom=0.4)
plt.title("Feature importances\nHapiness model before clustering")
plt.bar(range(n_features), importances[indices][:n_features],
        color="r", yerr=std[indices[:n_features]], align="center")
plt.xticks(range(n_features), features_short_name_sorted[:n_features], rotation=90)
plt.xlim([-1, n_features])
filename = path_fig / Path("feature_importance_option2.jpg")
plt.savefig(filename, format='jpg')
plt.show()

for f in range(min(X.shape[1],n_features_max)):
    print("%d. feature %d -%s- (%f)" % (f + 1, indices[f],features_name[indices[f]], im
    if features_name[indices[f]] in actionable_individual_1_features:
        print("\tActionable at individual level (1)")
    if features_name[indices[f]] in actionable_individual_2_features:
        print("\tActionable at individual level (2)")
    if features_name[indices[f]] in actionable_admin_1_features:
        print("\tActionable at administrative level (1)")
    if features_name[indices[f]] in actionable_admin_2_features:
        print("\tActionable at administrative level (2)")
```

```
Feature ranking:
```



```
1. feature 13 -recreation_score- (0.059650)
2. feature 21 -NIVPERSO- (0.059020)
        Actionable at individual level (2)
        Actionable at administrative level (2)
3. feature 28 -NOT_AMIS- (0.049001)
        Actionable at individual level (1)
        Actionable at administrative level (2)
4. feature 11 -service_score- (0.046573)
5. feature 19 -NOT_LIBR- (0.045039)
        Actionable at individual level (1)
        Actionable at administrative level (1)
6. feature 26 -health_score- (0.044641)
7. feature 12 -SECURITE- (0.041818)
        Actionable at individual level (2)
        Actionable at administrative level (1)
8. feature 10 -CADVIE- (0.041653)
        Actionable at individual level (1)
        Actionable at administrative level (1)
9. feature 2 -ETATSAN- (0.040500)
        Actionable at individual level (1)
        Actionable at administrative level (1)
10. feature 27 -SOUFFDEP_Oui- (0.040179)
        Actionable at individual level (2)
        Actionable at administrative level (1)
11. feature 9 -NOT_FAMI- (0.038618)
        Actionable at individual level (1)
        Actionable at administrative level (2)
12. feature 1 -INQALIM- (0.038176)
        Actionable at individual level (1)
        Actionable at administrative level (1)
13. feature 23 -score_environnement- (0.037665)
14. feature 18 -justice_score- (0.033420)
15. feature 32 -senior_score- (0.030732)
16. feature 30 -disability_score- (0.027630)
17. feature 8 -SOUFFINS_Oui- (0.025481)
        Actionable at individual level (2)
        Actionable at administrative level (1)
18. feature 3 -SITUEMP3_Inactif- (0.024076)
        Actionable at individual level (2)
        Actionable at administrative level (2)
19. feature 17 -RESTRICT_Oui- (0.024073)
        Actionable at individual level (2)
        Actionable at administrative level (2)
20. feature 16 -score_zone_humide- (0.022225)
21. feature 22 -score_nature- (0.022072)
22. feature 31 -SOUFFNER_Oui- (0.021519)
        Actionable at individual level (2)
        Actionable at administrative level (1)
```

```
23. feature 33 -rehabilitation_score- (0.021483)
24. feature 15 -child_score- (0.020440)
25. feature 20 -VACANCES_Oui- (0.019738)
        Actionable at individual level (1)
        Actionable at administrative level (1)
```

```
In [24]: print(f"number of features : {len(scope)} ow actionnable")
         A = scope & dict_features_sets.get('cdv_actionable_admin_1_features',set())
         B = scope & dict_features_sets.get('cdv_actionable_individual_1_features',set())
         print(f"- at administrative level 1 : \t{len(A)}\n- at individual level 1 : \t{len(B)}"
         important_features = set(features_name[indices][:10])
         C = A & important_features
         D = B & important_features
         print(f"- at administrative level 1 in top 10: \t{len(C)}\n- at individual level 1 in t
```

```
number of features : 34 ow actionnable
- at administrative level 1 :         11
- at individual level 1 :         9
- at administrative level 1 in top 10:         5
- at individual level 1 in top 10:         4
```

```
In [25]: n_estimators_range = [16,32,64,128]
         max_depth_range = [2,4,8,16,32,64]
         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_weight' : 'balanced'
                  }
         #scope = ( SelectFromModel_LogisticRegression_features )  & set(dataset.columns)
         features = df.loc[:,scope].columns
```

```
In [26]: score_clustering_methods = []
         clustering_methods = clustTest1.columns[2:3]

         for method in clustering_methods:
             print("--------------------------------------------")
             print(f"\nAnalysis cluster method {method}")
             cluster_list = clustTest1[method].unique()
             print(f"liste of clusters : {cluster_list}")
             score_cluster = []
             for cluster in cluster_list:
                 index_scope = clustTest1.loc[clustTest1[method]==cluster,:].index
                 print(f"cluster {cluster} '{cluster_name[cluster]}' : {len(index_scope)} elemen

                 Xc = X.loc[index_scope.intersection(X.index),:]
                 yc = y[index_scope.intersection(X.index)]
```

```
        Xs, ys = resample(Xc, yc, random_state=42)

        Xs = Xs.iloc[0:n_max,:]
        ys = ys.iloc[0:n_max]

        X_train, X_test, y_train, y_test = train_test_split(Xs, ys,
                                                            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: {ys.shape[0]}\n\
        - training set: {y_train.shape[0]}\n\
        - test set: {y_test.shape[0]}")
        print(f"Number of features: p={X_train.shape[1]}")
        print(f"Number of class: {len(np.unique(y))}")
        for c in np.unique(y):
            print(f"class {c:0.0f} : {100*np.sum(yc==c)/len(yc):0.1f}%")


        startTime = time.time()
        clf = RandomForestClassifier(**params)
        grid = GridSearchCV(clf,
                            scoring='accuracy',
                            param_grid=param_grid)

        grid.fit(X_train, y_train)
        print(f"Optimal values are {grid.best_params_} \n\
cross validation score {100*grid.best_score_:0.2f}%")
        print()

        # Learning on full training set with optimals hyperparameters and score on test
        params_opt = {'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_opt).fit(X_train, y_train)


        y_test_pred = clf.predict(X_test)
        accuracy = clf.score(X_test, y_test)
        f1 = f1_score(y_test, y_test_pred)
        p = precision_score(y_test, y_test_pred)
        r = recall_score(y_test, y_test_pred)
```

```
                res  = {'f1_score' : f1,
                        'accuracy' : accuracy,
                        'precision' : p,
                        'recall' : r}


                importances = clf.feature_importances_
                std = np.std([tree.feature_importances_ for tree in clf.estimators_],
                             axis=0)
                indices = np.argsort(importances)[::-1]
                features_name = np.array(features)

                cl = {'cluster' : cluster,
                      'size' : len(index_scope),
                      'model' : 'RandomForestClassifier',
                      'params' : params_opt,
                      'metrics' : res,
                      'importances' : importances,
                      'sdt' : std,
                      'indices' : indices,
                      'features_name' : features_name
                     }

                score_cluster.append(cl)


            d = {'clustering_method' : method,
                 'cluster_scores' : score_cluster
                }
            score_clustering_methods.append(d)
```

```
--------------------------------------------

Analysis cluster method clust3
liste of clusters : [2 4 6 1 3 5]
cluster 2 'Equilibrés' : 3053 elements
Number exemple: 2000
        - training set: 1600
        - test set: 400
Number of features: p=34
Number of class: 2
class 0 : 34.7%
class 1 : 65.3%
Optimal values are {'max_depth': 16, 'n_estimators': 128}
cross validation score 81.69%

cluster 4 'Domestiques modérés' : 2359 elements
Number exemple: 2000
```

```
        - training set: 1600
        - test set: 400
Number of features: p=34
Number of class: 2
class 0 : 32.7%
class 1 : 67.3%
Optimal values are {'max_depth': 32, 'n_estimators': 128}
cross validation score 84.50%

cluster 6 'Domestiques stricts' : 2313 elements
Number exemple: 2000
        - training set: 1600
        - test set: 400
Number of features: p=34
Number of class: 2
class 0 : 32.7%
class 1 : 67.3%
Optimal values are {'max_depth': 16, 'n_estimators': 64}
cross validation score 83.56%

cluster 1 'Civiques' : 528 elements
Number exemple: 514
        - training set: 411
        - test set: 103
Number of features: p=34
Number of class: 2
class 0 : 48.2%
class 1 : 51.8%
Optimal values are {'max_depth': 16, 'n_estimators': 64}
cross validation score 82.00%

cluster 3 'Flâneurs' : 1384 elements
Number exemple: 1367
        - training set: 1093
        - test set: 274
Number of features: p=34
Number of class: 2
class 0 : 31.6%
class 1 : 68.4%
Optimal values are {'max_depth': 16, 'n_estimators': 64}
cross validation score 85.82%

cluster 5 'Solitaires' : 1494 elements
Number exemple: 1482
        - training set: 1185
        - test set: 297
Number of features: p=34
Number of class: 2
```

```
class 0 : 40.8%
class 1 : 59.2%
Optimal values are {'max_depth': 32, 'n_estimators': 128}
cross validation score 84.98%
```

```
In [27]: # F1 score
         for score_method in score_clustering_methods:
             print(f"method {score_method['clustering_method']}:")
             average_score = 0
             total_size = 0
             for i, score_cluster in enumerate(score_method['cluster_scores']):
                 print(f"cluster {score_cluster['cluster']} '{cluster_name[score_cluster['cluste
                 average_score += score_cluster['metrics']['f1_score']*score_cluster['size']
                 total_size += score_cluster['size']

             average_score = average_score / total_size
             print(f"average f1 on clusters {100*average_score:0.1f}% gain {100*(average_score-r
```

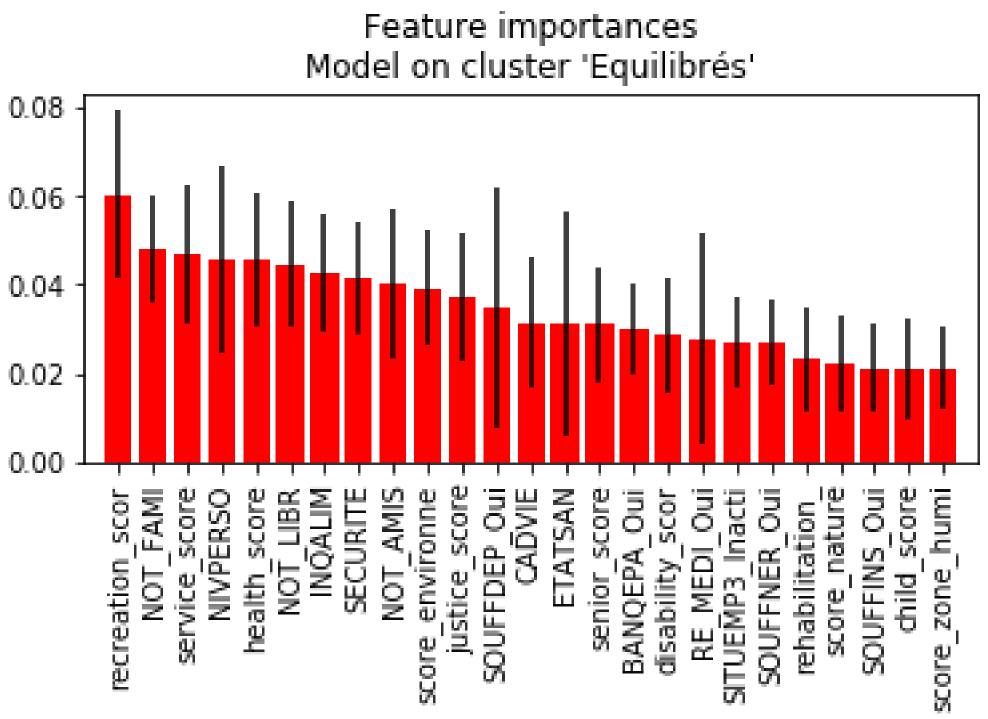
```
method clust3:
cluster 2 'Equilibrés' : (3053), f1 macro 87.4%
cluster 4 'Domestiques modérés' : (2359), f1 macro 89.7%
cluster 6 'Domestiques stricts' : (2313), f1 macro 92.5%
cluster 1 'Civiques' : (528), f1 macro 86.0%
cluster 3 'Flâneurs' : (1384), f1 macro 93.0%
cluster 5 'Solitaires' : (1494), f1 macro 89.5%
average f1 on clusters 89.9% gain 8.4
```

```
In [28]: # accuracy
         for score_method in score_clustering_methods:
             print(f"method {score_method['clustering_method']}:")
             average_score = 0
             total_size = 0
             for i, score_cluster in enumerate(score_method['cluster_scores']):
                 print(f"cluster {score_cluster['cluster']} '{cluster_name[score_cluster['cluste
                 average_score = average_score + score_cluster['metrics']['accuracy']*score_clus
                 total_size += score_cluster['size']
             average_score = average_score / total_size
             print(f"average accuracy on clusters {100*average_score:0.1f}% gain {100*(average_s
```

```
method clust3:
cluster 2 'Equilibrés' : (3053) accuracy 83.8%
cluster 4 'Domestiques modérés' : (2359) accuracy 86.0%
cluster 6 'Domestiques stricts' : (2313) accuracy 89.2%
cluster 1 'Civiques' : (528) accuracy 83.5%
cluster 3 'Flâneurs' : (1384) accuracy 89.8%
cluster 5 'Solitaires' : (1494) accuracy 88.6%
average accuracy on clusters 86.8% gain 13.3
```

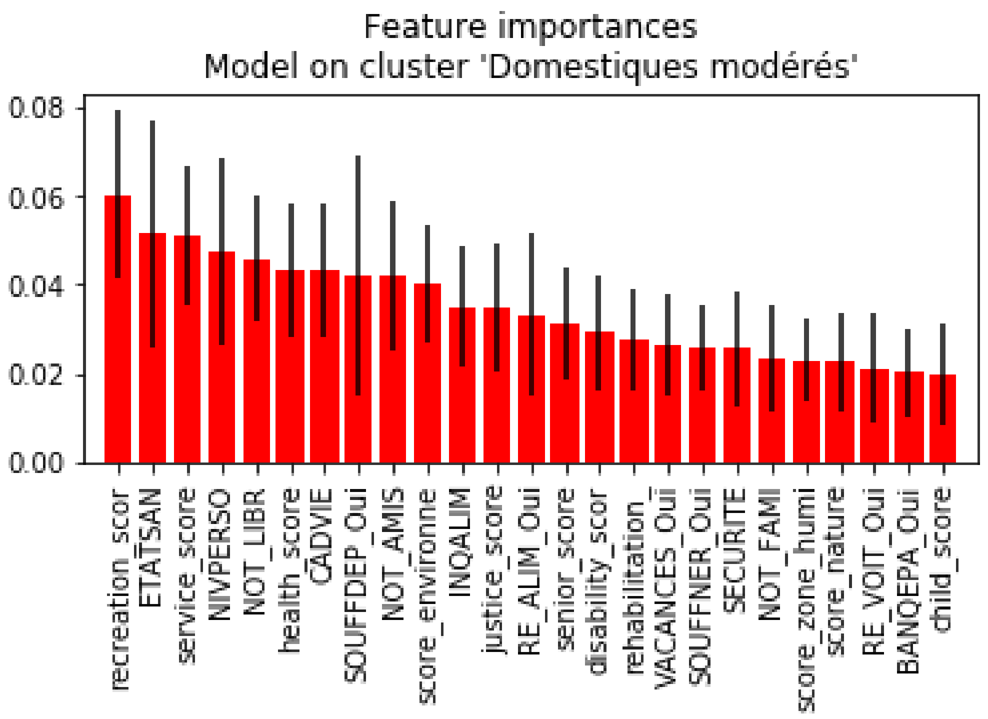
```
In [29]: # Feature importance by cluster
         for score_method in score_clustering_methods:
             print(f"method {score_method['clustering_method']}:")
             for i, score_cluster in enumerate(score_method['cluster_scores']):
                 print(f"cluster {score_cluster['cluster']} ({score_cluster['size']}), f1 macro
                 print(f"top 15 features:")
                 indices = score_cluster['indices']
                 features_name = score_cluster['features_name']
                 importances = score_cluster['importances']
                 features_short_name_sorted = [ name[:15] for name in features_name[indices]]

                 # Plot the feature importances of the forest
                 plt.figure()
                 plt.gcf().subplots_adjust(bottom=0.4)
                 plt.title(f"Feature importances\nModel on cluster '{cluster_name[score_cluster[
                 plt.bar(range(n_features), importances[indices][:n_features],
                         color="r", yerr=std[indices][:n_features], align="center")
                 plt.xticks(range(n_features), features_short_name_sorted[:n_features], rotation
                 plt.xlim([-1, n_features])
                 filename = path_fig / Path(f"feature_importance_cluster_{score_cluster['cluster
                 plt.savefig(filename, format='jpg')
                 plt.show()

                 for f in range(n_features_max):
                     print("%d. feature %d -%s- (%f)" % (f + 1, indices[f],features_name[indices
                     if features_name[indices[f]] in actionable_individual_1_features:
                         print("\tActionable at individual level (1)")
                     if features_name[indices[f]] in actionable_individual_2_features:
                         print("\tActionable at individual level (2)")
                     if features_name[indices[f]] in actionable_admin_1_features:
                         print("\tActionable at administrative level (1)")
                     if features_name[indices[f]] in actionable_admin_2_features:
                         print("\tActionable at administrative level (2)")
```

```
method clust3:
cluster 2 (3053), f1 macro 87.4%
top 15 features:
```

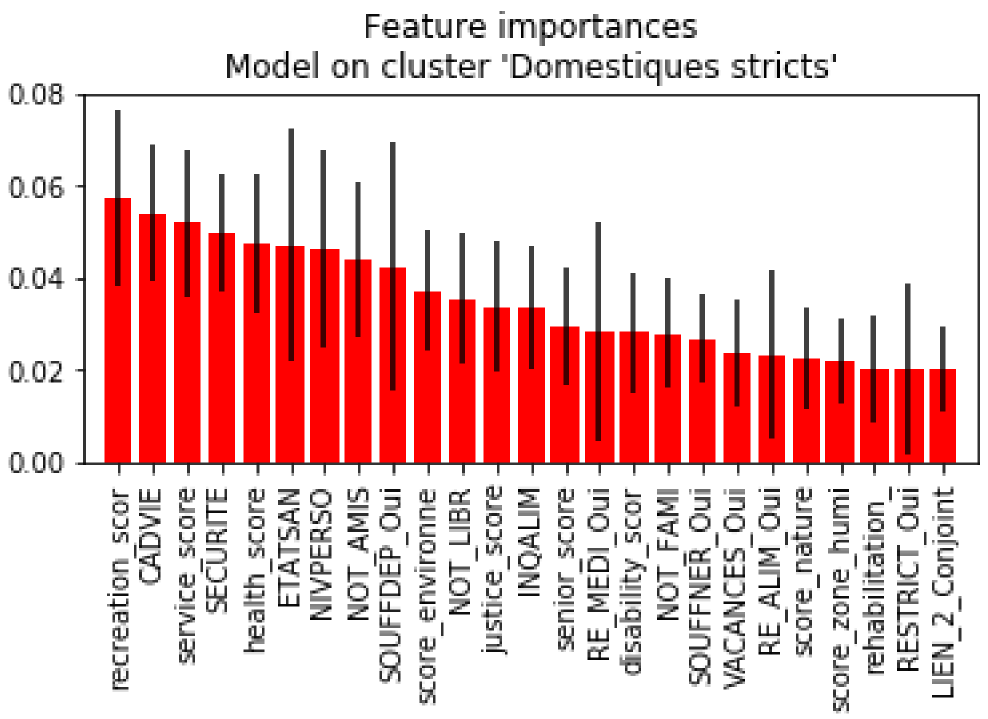


```
1. feature 13 -recreation_score- (0.060240)
2. feature 9 -NOT_FAMI- (0.048089)
        Actionable at individual level (1)
        Actionable at administrative level (2)
3. feature 11 -service_score- (0.046739)
4. feature 21 -NIVPERSO- (0.045620)
        Actionable at individual level (2)
        Actionable at administrative level (2)
5. feature 26 -health_score- (0.045372)
6. feature 19 -NOT_LIBR- (0.044479)
        Actionable at individual level (1)
        Actionable at administrative level (1)
7. feature 1 -INQALIM- (0.042796)
        Actionable at individual level (1)
        Actionable at administrative level (1)
8. feature 12 -SECURITE- (0.041377)
        Actionable at individual level (2)
        Actionable at administrative level (1)
9. feature 28 -NOT_AMIS- (0.040178)
        Actionable at individual level (1)
        Actionable at administrative level (2)
10. feature 23 -score_environnement- (0.039250)
11. feature 18 -justice_score- (0.037153)
12. feature 27 -SOUFFDEP_Oui- (0.034893)
```

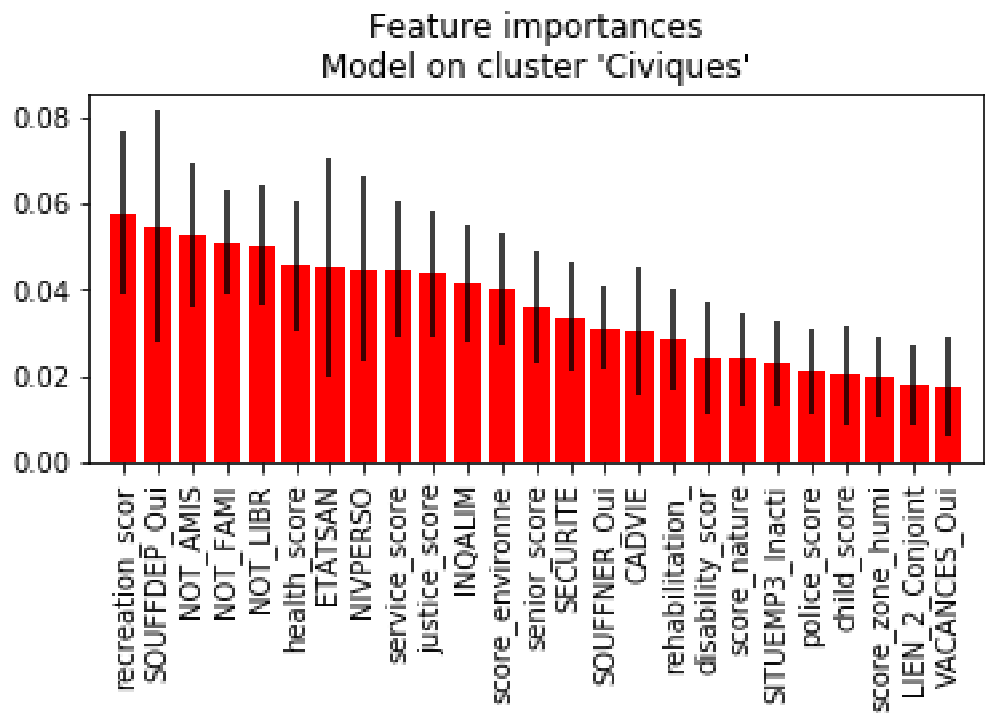
```
        Actionable at individual level (2)
        Actionable at administrative level (1)
13. feature 10 -CADVIE- (0.031316)
        Actionable at individual level (1)
        Actionable at administrative level (1)
14. feature 2 -ETATSAN- (0.031159)
        Actionable at individual level (1)
        Actionable at administrative level (1)
15. feature 32 -senior_score- (0.030968)
16. feature 14 -BANQEPA_Oui- (0.029996)
        Actionable at individual level (1)
        Actionable at administrative level (1)
17. feature 30 -disability_score- (0.028602)
18. feature 24 -RE_MEDI_Oui- (0.027684)
        Actionable at individual level (2)
        Actionable at administrative level (2)
19. feature 3 -SITUEMP3_Inactif- (0.027015)
        Actionable at individual level (2)
        Actionable at administrative level (2)
20. feature 31 -SOUFFNER_Oui- (0.026765)
        Actionable at individual level (2)
        Actionable at administrative level (1)
21. feature 33 -rehabilitation_score- (0.023318)
22. feature 22 -score_nature- (0.022155)
23. feature 8 -SOUFFINS_Oui- (0.021280)
        Actionable at individual level (2)
        Actionable at administrative level (1)
24. feature 15 -child_score- (0.021161)
25. feature 16 -score_zone_humide- (0.021134)
cluster 4 (2359), f1 macro 89.7%
top 15 features:
```



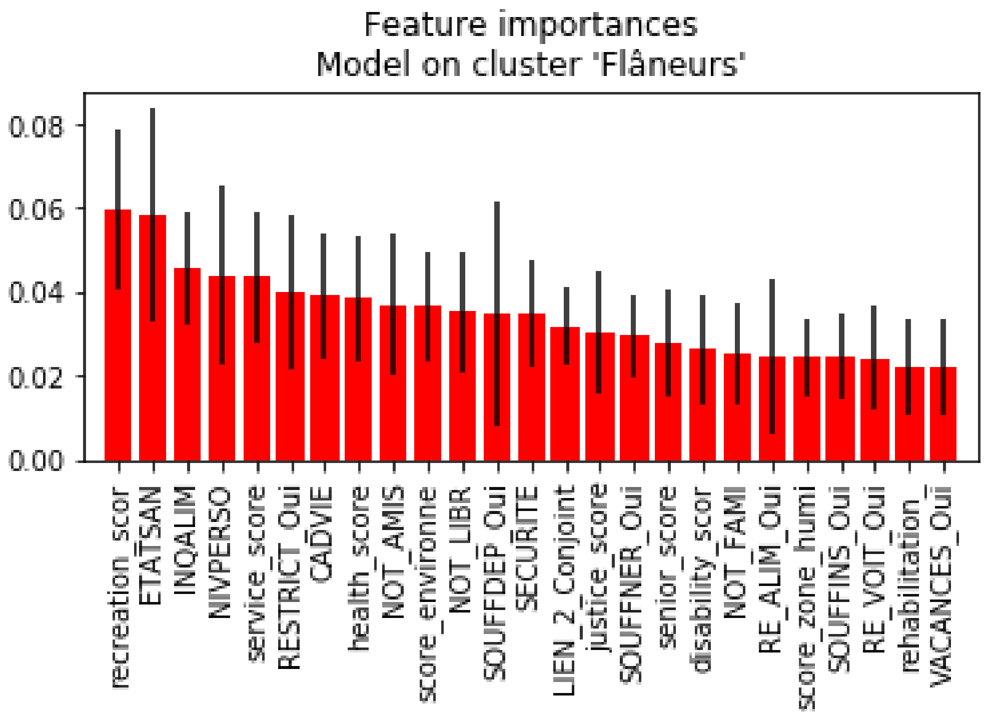
```
1. feature 13 -recreation_score- (0.060207)
2. feature 2 -ETATSAN- (0.051485)
        Actionable at individual level (1)
        Actionable at administrative level (1)
3. feature 11 -service_score- (0.050973)
4. feature 21 -NIVPERSO- (0.047533)
        Actionable at individual level (2)
        Actionable at administrative level (2)
5. feature 19 -NOT_LIBR- (0.045649)
        Actionable at individual level (1)
        Actionable at administrative level (1)
6. feature 26 -health_score- (0.043495)
7. feature 10 -CADVIE- (0.043246)
        Actionable at individual level (1)
        Actionable at administrative level (1)
8. feature 27 -SOUFFDEP_Oui- (0.042098)
        Actionable at individual level (2)
        Actionable at administrative level (1)
9. feature 28 -NOT_AMIS- (0.041891)
        Actionable at individual level (1)
        Actionable at administrative level (2)
10. feature 23 -score_environnement- (0.040035)
11. feature 1 -INQALIM- (0.035039)
        Actionable at individual level (1)
        Actionable at administrative level (1)
12. feature 18 -justice_score- (0.034880)
13. feature 0 -RE_ALIM_Oui- (0.033203)
        Actionable at individual level (2)
        Actionable at administrative level (2)
14. feature 32 -senior_score- (0.031288)
15. feature 30 -disability_score- (0.029228)
16. feature 33 -rehabilitation_score- (0.027566)
17. feature 20 -VACANCES_Oui- (0.026552)
        Actionable at individual level (1)
        Actionable at administrative level (1)
18. feature 31 -SOUFFNER_Oui- (0.025747)
        Actionable at individual level (2)
        Actionable at administrative level (1)
19. feature 12 -SECURITE- (0.025563)
        Actionable at individual level (2)
        Actionable at administrative level (1)
20. feature 9 -NOT_FAMI- (0.023414)
        Actionable at individual level (1)
        Actionable at administrative level (2)
21. feature 16 -score_zone_humide- (0.023009)
22. feature 22 -score_nature- (0.022681)
23. feature 4 -RE_VOIT_Oui- (0.021207)
        Actionable at individual level (2)
        Actionable at administrative level (2)
24. feature 14 -BANQEPA_Oui- (0.020110)
        Actionable at individual level (1)
        Actionable at administrative level (1)
25. feature 15 -child_score- (0.019883)
cluster 6 (2313), f1 macro 92.5%
top 15 features:
```



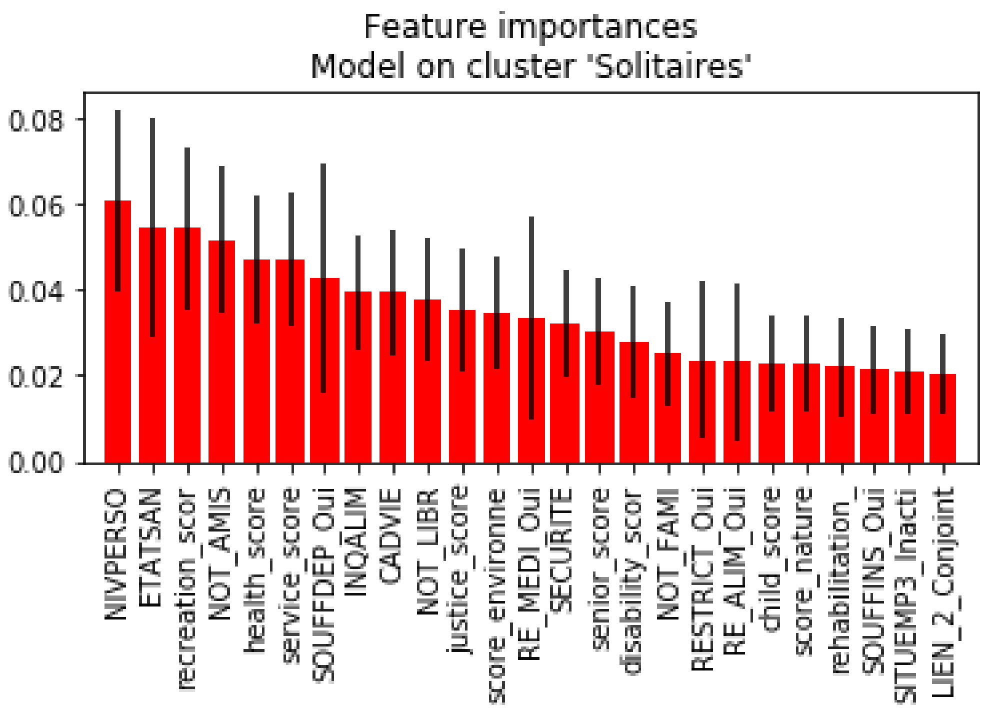
```
1. feature 13 -recreation_score- (0.057281)
2. feature 10 -CADVIE- (0.053844)
        Actionable at individual level (1)
        Actionable at administrative level (1)
3. feature 11 -service_score- (0.051812)
4. feature 12 -SECURITE- (0.049728)
        Actionable at individual level (2)
        Actionable at administrative level (1)
5. feature 26 -health_score- (0.047156)
6. feature 2 -ETATSAN- (0.047098)
        Actionable at individual level (1)
        Actionable at administrative level (1)
7. feature 21 -NIVPERSO- (0.046260)
        Actionable at individual level (2)
        Actionable at administrative level (2)
8. feature 28 -NOT_AMIS- (0.043982)
        Actionable at individual level (1)
        Actionable at administrative level (2)
9. feature 27 -SOUFFDEP_Oui- (0.042436)
        Actionable at individual level (2)
        Actionable at administrative level (1)
10. feature 23 -score_environnement- (0.037200)
11. feature 19 -NOT_LIBR- (0.035333)
        Actionable at individual level (1)
        Actionable at administrative level (1)
12. feature 18 -justice_score- (0.033776)
13. feature 1 -INQALIM- (0.033651)
        Actionable at individual level (1)
        Actionable at administrative level (1)
14. feature 32 -senior_score- (0.029387)
15. feature 24 -RE_MEDI_Oui- (0.028344)
        Actionable at individual level (2)
        Actionable at administrative level (2)
16. feature 30 -disability_score- (0.028124)
17. feature 9 -NOT_FAMI- (0.027908)
        Actionable at individual level (1)
        Actionable at administrative level (2)
18. feature 31 -SOUFFNER_Oui- (0.026827)
        Actionable at individual level (2)
        Actionable at administrative level (1)
19. feature 20 -VACANCES_Oui- (0.023851)
        Actionable at individual level (1)
        Actionable at administrative level (1)
20. feature 0 -RE_ALIM_Oui- (0.023308)
        Actionable at individual level (2)
        Actionable at administrative level (2)
21. feature 22 -score_nature- (0.022559)
22. feature 16 -score_zone_humide- (0.022062)
23. feature 33 -rehabilitation_score- (0.020259)
24. feature 17 -RESTRICT_Oui- (0.020255)
        Actionable at individual level (2)
        Actionable at administrative level (2)
25. feature 5 -LIEN_2_Conjoint ou compagnon- (0.020177)
cluster 1 (528), f1 macro 86.0%
top 15 features:
```



```
1. feature 13 -recreation_score- (0.057651)
2. feature 27 -SOUFFDEP_Oui- (0.054654)
        Actionable at individual level (2)
        Actionable at administrative level (1)
3. feature 28 -NOT_AMIS- (0.052522)
        Actionable at individual level (1)
        Actionable at administrative level (2)
4. feature 9 -NOT_FAMI- (0.050816)
        Actionable at individual level (1)
        Actionable at administrative level (2)
5. feature 19 -NOT_LIBR- (0.050318)
        Actionable at individual level (1)
        Actionable at administrative level (1)
6. feature 26 -health_score- (0.045595)
7. feature 2 -ETATSAN- (0.044938)
        Actionable at individual level (1)
        Actionable at administrative level (1)
8. feature 21 -NIVPERSO- (0.044766)
        Actionable at individual level (2)
        Actionable at administrative level (2)
9. feature 11 -service_score- (0.044677)
10. feature 18 -justice_score- (0.043685)
11. feature 1 -INQALIM- (0.041439)
        Actionable at individual level (1)
        Actionable at administrative level (1)
12. feature 23 -score_environnement- (0.040245)
13. feature 32 -senior_score- (0.035830)
14. feature 12 -SECURITE- (0.033536)
        Actionable at individual level (2)
        Actionable at administrative level (1)
15. feature 31 -SOUFFNER_Oui- (0.030934)
        Actionable at individual level (2)
        Actionable at administrative level (1)
16. feature 10 -CADVIE- (0.030289)
        Actionable at individual level (1)
        Actionable at administrative level (1)
17. feature 33 -rehabilitation_score- (0.028356)
18. feature 30 -disability_score- (0.023987)
19. feature 22 -score_nature- (0.023846)
20. feature 3 -SITUEMP3_Inactif- (0.022788)
        Actionable at individual level (2)
        Actionable at administrative level (2)
21. feature 7 -police_score- (0.021120)
22. feature 15 -child_score- (0.020151)
23. feature 16 -score_zone_humide- (0.019625)
24. feature 5 -LIEN_2_Conjoint ou compagnon- (0.017995)
25. feature 20 -VACANCES_Oui- (0.017589)
        Actionable at individual level (1)
        Actionable at administrative level (1)
cluster 3 (1384), f1 macro 93.0%
top 15 features:
```



```
1. feature 13 -recreation_score- (0.059557)
2. feature 2 -ETATSAN- (0.058241)
        Actionable at individual level (1)
        Actionable at administrative level (1)
3. feature 1 -INQALIM- (0.045958)
        Actionable at individual level (1)
        Actionable at administrative level (1)
4. feature 21 -NIVPERSO- (0.043906)
        Actionable at individual level (2)
        Actionable at administrative level (2)
5. feature 11 -service_score- (0.043523)
6. feature 17 -RESTRICT_Oui- (0.040118)
        Actionable at individual level (2)
        Actionable at administrative level (2)
7. feature 10 -CADVIE- (0.039059)
        Actionable at individual level (1)
        Actionable at administrative level (1)
8. feature 26 -health_score- (0.038479)
9. feature 28 -NOT_AMIS- (0.037006)
        Actionable at individual level (1)
        Actionable at administrative level (2)
10. feature 23 -score_environnement- (0.036567)
11. feature 19 -NOT_LIBR- (0.035219)
        Actionable at individual level (1)
        Actionable at administrative level (1)
12. feature 27 -SOUFFDEP_Oui- (0.034982)
        Actionable at individual level (2)
        Actionable at administrative level (1)
13. feature 12 -SECURITE- (0.034640)
        Actionable at individual level (2)
        Actionable at administrative level (1)
14. feature 5 -LIEN_2_Conjoint ou compagnon- (0.031888)
15. feature 18 -justice_score- (0.030354)
16. feature 31 -SOUFFNER_Oui- (0.029586)
        Actionable at individual level (2)
        Actionable at administrative level (1)
17. feature 32 -senior_score- (0.027716)
18. feature 30 -disability_score- (0.026399)
19. feature 9 -NOT_FAMI- (0.025206)
        Actionable at individual level (1)
        Actionable at administrative level (2)
20. feature 0 -RE_ALIM_Oui- (0.024695)
        Actionable at individual level (2)
        Actionable at administrative level (2)
21. feature 16 -score_zone_humide- (0.024691)
22. feature 8 -SOUFFINS_Oui- (0.024533)
        Actionable at individual level (2)
        Actionable at administrative level (1)
23. feature 4 -RE_VOIT_Oui- (0.024299)
        Actionable at individual level (2)
        Actionable at administrative level (2)
24. feature 33 -rehabilitation_score- (0.022264)
25. feature 20 -VACANCES_Oui- (0.021980)
        Actionable at individual level (1)
        Actionable at administrative level (1)
cluster 5 (1494), f1 macro 89.5%
top 15 features:
```



```
1. feature 21 -NIVPERSO- (0.060923)
        Actionable at individual level (2)
        Actionable at administrative level (2)
2. feature 2 -ETATSAN- (0.054585)
        Actionable at individual level (1)
        Actionable at administrative level (1)
3. feature 13 -recreation_score- (0.054377)
4. feature 28 -NOT_AMIS- (0.051651)
        Actionable at individual level (1)
        Actionable at administrative level (2)
5. feature 26 -health_score- (0.047141)
6. feature 11 -service_score- (0.046994)
7. feature 27 -SOUFFDEP_Oui- (0.042860)
        Actionable at individual level (2)
        Actionable at administrative level (1)
8. feature 1 -INQALIM- (0.039438)
        Actionable at individual level (1)
        Actionable at administrative level (1)
9. feature 10 -CADVIE- (0.039396)
        Actionable at individual level (1)
        Actionable at administrative level (1)
10. feature 19 -NOT_LIBR- (0.037676)
        Actionable at individual level (1)
        Actionable at administrative level (1)
```

```
11. feature 18 -justice_score- (0.035045)
12. feature 23 -score_environnement- (0.034700)
13. feature 24 -RE_MEDI_Oui- (0.033579)
        Actionable at individual level (2)
        Actionable at administrative level (2)
14. feature 12 -SECURITE- (0.032083)
        Actionable at individual level (2)
        Actionable at administrative level (1)
15. feature 32 -senior_score- (0.030337)
16. feature 30 -disability_score- (0.027743)
17. feature 9 -NOT_FAMI- (0.024980)
        Actionable at individual level (1)
        Actionable at administrative level (2)
18. feature 17 -RESTRICT_Oui- (0.023518)
        Actionable at individual level (2)
        Actionable at administrative level (2)
19. feature 0 -RE_ALIM_Oui- (0.023200)
        Actionable at individual level (2)
        Actionable at administrative level (2)
20. feature 15 -child_score- (0.022877)
21. feature 22 -score_nature- (0.022672)
22. feature 33 -rehabilitation_score- (0.021916)
23. feature 8 -SOUFFINS_Oui- (0.021292)
        Actionable at individual level (2)
        Actionable at administrative level (1)
24. feature 3 -SITUEMP3_Inactif- (0.020953)
        Actionable at individual level (2)
        Actionable at administrative level (2)
25. feature 5 -LIEN_2_Conjoint ou compagnon- (0.020330)
```

1.3 Option 3

```
In [30]: # choosing set of features
         scope_filter = RFE_RandomForestClassifier_100_features | RFE_LinearSVC_100_features
         scope = dict_features_sets.get('insee_environment_score_features', set())
         scope = scope | dict_features_sets.get('insee_recreation_score_features',set())
         scope = scope | dict_features_sets.get('cdv_actionable_admin_1_features',set())
         scope = scope | dict_features_sets.get('cdv_actionable_individual_1_features',set())
         scope = scope | RFE_LogisticRegression_20_features
         scope =  scope & scope_filter
         print(f"number of features : {len(scope)} ow actionable")
         A = scope & dict_features_sets.get('cdv_actionable_admin_1_features',set())
         B = scope & dict_features_sets.get('cdv_actionable_individual_1_features',set())
         print(f"- at administrative level 1 : \t{len(A)}\n- at individual level 1 : \t{len(B)}"
```

```
number of features : 40 ow actionable
- at administrative level 1 :         29
- at individual level 1 :         26
```

```
In [31]: df = dataset.loc[:,:]
         # reducing problem to a 2 class classification problem
         df["HEUREUX_CLF"] = 0
         df.loc[df["HEUREUX"]==4, "HEUREUX_CLF"] = 1
         df.loc[df["HEUREUX"]==3, "HEUREUX_CLF"] = 1
         df.loc[df["HEUREUX"]==5, "HEUREUX_CLF"] = None

         scope = scope & set(dataset.columns)
         n_max = 2000

         df = df.loc[:,scope | {"HEUREUX_CLF"} ].dropna()
         features = df.loc[:,scope ].columns

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


         Xs, ys = resample(X, y, random_state=42)

         Xs = Xs.iloc[0:n_max,:]
         ys = ys.iloc[0:n_max]

         X_train, X_test, y_train, y_test = train_test_split(Xs, ys,
                                                              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: {y.shape[0]}\n- training set: \
         {y_train.shape[0]}\n- test set: {y_test.shape[0]}")
         print(f"Number of features: p={X_train.shape[1]}")
         print(f"Number of class: {len(np.unique(y))}")
         for c in np.unique(y):
             print(f"class {c:0.0f} : {100*np.sum(y==c)/len(y):0.1f}%")
```

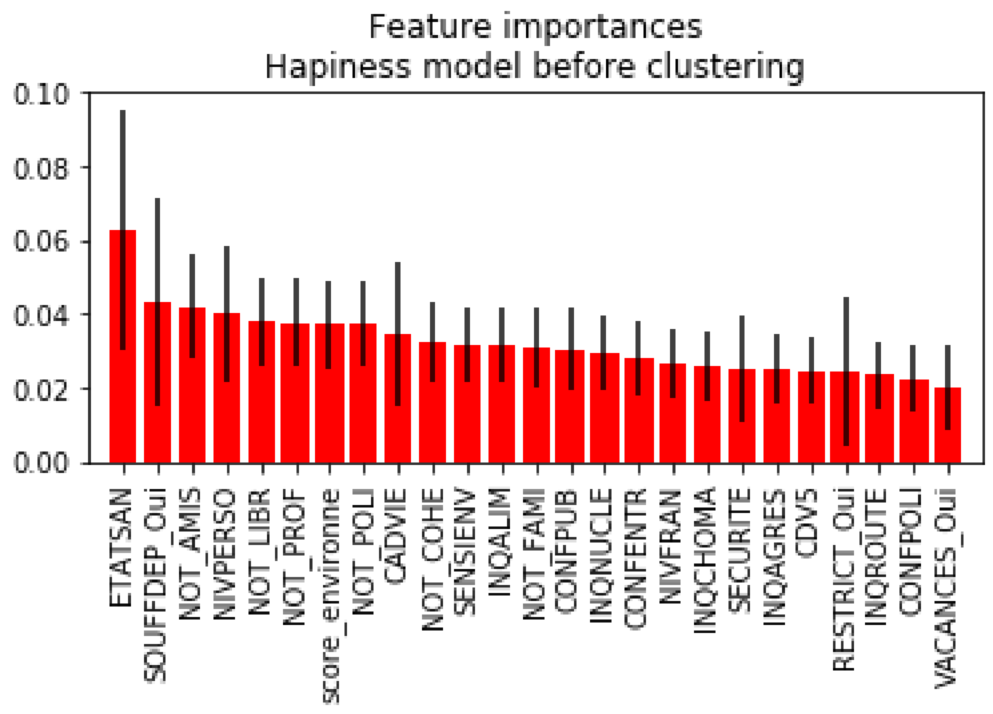
```
Number exemple: 10605
- training set: 1600
- test set: 400
Number of features: p=40
Number of class: 2
class 0 : 34.9%
class 1 : 65.1%
```

```
In [32]: startTime = time.time()
         n_estimators_range = [32,64,128,256,512]
         max_depth_range = [4,8,16,32,64]
         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_weight' : 'balanced'}
         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_} \n\
         Accuracy Score of cross valdation {100*grid.best_score_:0.2f}%")

         # 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)
         clf.fit(X_train, y_train)
         y_test_pred = clf.predict(X_test)

         print(f"Random Forest, p={X_train.shape[1]}")
         accuracy = clf.score(X_test, y_test)
         f1 = f1_score(y_test, y_test_pred)
         p = precision_score(y_test, y_test_pred)
         r = recall_score(y_test, y_test_pred)
         print(f"Model score\n- Accuracy : {accuracy*100:0.1f} %")
         print(f"- Precision : {p*100:0.1f} % (Happy # positive class)")
         print(f"- Recall : {r*100:0.1f} %")
         print(f"- F1 score : {f1*100:0.1f} %")
         res_full  = {
             'f1_score' : f1,
             'accuracy' : accuracy,
             'precision' : p,
             'recall' : r
         }
```

```
Determination of optimal hyperparameters in 61.3 s
Optimal values are {'max_depth': 16, 'n_estimators': 512}
Accuracy Score of cross valdation 75.88%
Random Forest, p=40
Model score
- Accuracy : 73.0 %
- Precision : 75.3 % (Happy # positive class)
- Recall : 85.4 %
- F1 score : 80.0 %
```

```
In [33]: importances = clf.feature_importances_
         std = np.std([tree.feature_importances_ for tree in clf.estimators_],
                      axis=0)
         indices = np.argsort(importances)[::-1]
         features_name = np.array(features)
         features_short_name_sorted = [ name[:15] for name in features_name[indices]]
         print("Feature ranking:")

         n_features_max = 25
         n_features = min(X.shape[1],n_features_max)

         # Plot the feature importances of the forest

         plt.figure()
         plt.gcf().subplots_adjust(bottom=0.4)
         plt.title("Feature importances\nHapiness model before clustering")
         plt.bar(range(n_features), importances[indices][:n_features],
                 color="r", yerr=std[indices[:n_features]], align="center")
         plt.xticks(range(n_features), features_short_name_sorted[:n_features], rotation=90)
         plt.xlim([-1, n_features])
         filename = path_fig / Path("feature_importance_option3.jpg")
         plt.savefig(filename, format='jpg')
         plt.show()

         for f in range(min(X.shape[1],n_features_max)):
             print("%d. feature %d -%s- (%f)" % (f + 1, indices[f],features_name[indices[f]], im
             if features_name[indices[f]] in actionable_individual_1_features:
                 print("\tActionable at individual level (1)")
             if features_name[indices[f]] in actionable_individual_2_features:
                 print("\tActionable at individual level (2)")
             if features_name[indices[f]] in actionable_admin_1_features:
                 print("\tActionable at administrative level (1)")
             if features_name[indices[f]] in actionable_admin_2_features:
                 print("\tActionable at administrative level (2)")
```

```
Feature ranking:
```



```
1. feature 3 -ETATSAN- (0.062689)
        Actionable at individual level (1)
        Actionable at administrative level (1)
2. feature 35 -SOUFFDEP_Oui- (0.042963)
        Actionable at individual level (2)
        Actionable at administrative level (1)
3. feature 36 -NOT_AMIS- (0.041908)
        Actionable at individual level (1)
        Actionable at administrative level (2)
4. feature 27 -NIVPERSO- (0.039819)
        Actionable at individual level (2)
        Actionable at administrative level (2)
5. feature 24 -NOT_LIBR- (0.037761)
        Actionable at individual level (1)
        Actionable at administrative level (1)
6. feature 26 -NOT_PROF- (0.037372)
        Actionable at individual level (1)
        Actionable at administrative level (2)
7. feature 28 -score_environnement- (0.037163)
8. feature 33 -NOT_POLI- (0.037133)
        Actionable at individual level (1)
        Actionable at administrative level (1)
9. feature 16 -CADVIE- (0.034626)
        Actionable at individual level (1)
        Actionable at administrative level (1)
10. feature 31 -NOT_COHE- (0.032463)
        Actionable at individual level (1)
        Actionable at administrative level (1)
11. feature 32 -SENSIENV- (0.031718)
        Actionable at individual level (1)
        Actionable at administrative level (1)
12. feature 2 -INQALIM- (0.031470)
        Actionable at individual level (1)
        Actionable at administrative level (1)
13. feature 14 -NOT_FAMI- (0.030716)
        Actionable at individual level (1)
        Actionable at administrative level (2)
14. feature 6 -CONFPUB- (0.030303)
        Actionable at individual level (1)
        Actionable at administrative level (1)
15. feature 39 -INQNUCLE- (0.029431)
        Actionable at individual level (1)
        Actionable at administrative level (1)
16. feature 21 -CONFENTR- (0.027902)
        Actionable at individual level (1)
        Actionable at administrative level (1)
17. feature 8 -NIVFRAN- (0.026394)
        Actionable at individual level (2)
        Actionable at administrative level (1)
18. feature 20 -INQCHOMA- (0.025716)
        Actionable at individual level (1)
        Actionable at administrative level (1)
19. feature 17 -SECURITE- (0.025106)
        Actionable at individual level (2)
        Actionable at administrative level (1)
20. feature 9 -INQAGRES- (0.024943)
        Actionable at individual level (1)
        Actionable at administrative level (1)
21. feature 7 -CDV5- (0.024466)
        Actionable at individual level (2)
        Actionable at administrative level (1)
22. feature 23 -RESTRICT_Oui- (0.023934)
        Actionable at individual level (2)
        Actionable at administrative level (2)
23. feature 11 -INQROUTE- (0.023345)
        Actionable at individual level (1)
        Actionable at administrative level (1)
24. feature 30 -CONFPOLI- (0.022245)
        Actionable at individual level (1)
        Actionable at administrative level (1)
25. feature 25 -VACANCES_Oui- (0.020232)
        Actionable at individual level (1)
        Actionable at administrative level (1)
```

```
In [34]: print(f"number of features : {len(scope)} ow actionnable")
         A = scope & dict_features_sets.get('cdv_actionable_admin_1_features',set())
         B = scope & dict_features_sets.get('cdv_actionable_individual_1_features',set())
         print(f"- at administrative level 1 : \t{len(A)}\n- at individual level 1 : \t{len(B)}"
         important_features = set(features_name[indices][:10])
         C = A & important_features
         D = B & important_features
         print(f"- at administrative level 1 in top 10: \t{len(C)}\n- at individual level 1 in t
```

```
number of features : 40 ow actionnable
- at administrative level 1 :         29
- at individual level 1 :         26
- at administrative level 1 in top 10:         6
- at individual level 1 in top 10:         7
```

```
In [35]: n_estimators_range = [16,32,64,128]
         max_depth_range = [2,4,8,16,32,64]
         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_weight' : 'balanced'
                  }
         #scope = ( SelectFromModel_LogisticRegression_features )  & set(dataset.columns)
         features = df.loc[:,scope].columns
```

```
In [36]: score_clustering_methods = []
         clustering_methods = clustTest1.columns[2:3]

         for method in clustering_methods:
             print("---------------------------------------------")
             print(f"\nAnalysis cluster method {method}")
             cluster_list = clustTest1[method].unique()
             print(f"liste of clusters : {cluster_list}")
             score_cluster = []
             for cluster in cluster_list:
                 index_scope = clustTest1.loc[clustTest1[method]==cluster,:].index
                 print(f"cluster {cluster} '{cluster_name[cluster]}' : {len(index_scope)} elemen

                 Xc = X.loc[index_scope.intersection(X.index),:]
                 yc = y[index_scope.intersection(X.index)]

                 Xs, ys = resample(Xc, yc, random_state=42)

                 Xs = Xs.iloc[0:n_max,:]
```

```
        ys = ys.iloc[0:n_max]

        X_train, X_test, y_train, y_test = train_test_split(Xs, ys,
                                                            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: {ys.shape[0]}\n\
- training set: {y_train.shape[0]}\n\
- test set: {y_test.shape[0]}")
        print(f"Number of features: p={X_train.shape[1]}")
        print(f"Number of class: {len(np.unique(y))}")
        for c in np.unique(y):
            print(f"class {c:0.0f} : {100*np.sum(yc==c)/len(yc):0.1f}%")


        startTime = time.time()
        clf = RandomForestClassifier(**params)
        grid = GridSearchCV(clf,
                            scoring='accuracy',
                            param_grid=param_grid)

        grid.fit(X_train, y_train)
        print(f"Optimal values are {grid.best_params_} \n\
cross validation score {100*grid.best_score_:0.2f}%")
        print()

        # Learning on full training set with optimals hyperparameters and score on test
        params_opt = {'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_opt).fit(X_train, y_train)


        y_test_pred = clf.predict(X_test)
        accuracy = clf.score(X_test, y_test)
        f1 = f1_score(y_test, y_test_pred)
        p = precision_score(y_test, y_test_pred)
        r = recall_score(y_test, y_test_pred)

        res  = {'f1_score' : f1,
                'accuracy' : accuracy,
                'precision' : p,
                'recall' : r}
        importances = clf.feature_importances_
        std = np.std([tree.feature_importances_ for tree in clf.estimators_],
                     axis=0)
        indices = np.argsort(importances)[::-1]
        features_name = np.array(features)

        cl = {'cluster' : cluster,
              'size' : len(index_scope),
              'model' : 'RandomForestClassifier',
              'params' : params_opt,
              'metrics' : res,
              'importances' : importances,
              'sdt' : std,
              'indices' : indices,
              'features_name' : features_name
             }

        score_cluster.append(cl)


    d = {'clustering_method' : method,
         'cluster_scores' : score_cluster
        }
    score_clustering_methods.append(d)
```

```
--------------------------------------------

Analysis cluster method clust3
liste of clusters : [2 4 6 1 3 5]
cluster 2 'Equilibrés' : 3053 elements
Number exemple: 2000
        - training set: 1600
        - test set: 400
Number of features: p=40
Number of class: 2
class 0 : 34.6%
class 1 : 65.4%
Optimal values are {'max_depth': 16, 'n_estimators': 64}
cross validation score 81.94%

cluster 4 'Domestiques modérés' : 2359 elements
Number exemple: 2000
        - training set: 1600
        - test set: 400
Number of features: p=40
Number of class: 2
```

```
class 0 : 32.6%
class 1 : 67.4%
Optimal values are {'max_depth': 16, 'n_estimators': 128}
cross validation score 83.94%

cluster 6 'Domestiques stricts' : 2313 elements
Number exemple: 2000
        - training set: 1600
        - test set: 400
Number of features: p=40
Number of class: 2
class 0 : 32.8%
class 1 : 67.2%
Optimal values are {'max_depth': 32, 'n_estimators': 128}
cross validation score 83.12%

cluster 1 'Civiques' : 528 elements
Number exemple: 468
        - training set: 374
        - test set: 94
Number of features: p=40
Number of class: 2
class 0 : 50.2%
class 1 : 49.8%
Optimal values are {'max_depth': 8, 'n_estimators': 64}
cross validation score 73.80%

cluster 3 'Flâneurs' : 1384 elements
Number exemple: 1350
        - training set: 1080
        - test set: 270
Number of features: p=40
Number of class: 2
class 0 : 31.7%
class 1 : 68.3%
Optimal values are {'max_depth': 16, 'n_estimators': 64}
cross validation score 85.37%

cluster 5 'Solitaires' : 1494 elements
Number exemple: 1447
        - training set: 1157
        - test set: 290
Number of features: p=40
Number of class: 2
class 0 : 40.6%
class 1 : 59.4%
Optimal values are {'max_depth': 32, 'n_estimators': 64}
cross validation score 84.44%
```

```
In [37]: # F1 score
         for score_method in score_clustering_methods:
             print(f"method {score_method['clustering_method']}:")
             average_score = 0
             total_size = 0
             for i, score_cluster in enumerate(score_method['cluster_scores']):
                 print(f"cluster {score_cluster['cluster']} '{cluster_name[score_cluster['cluste
                 average_score += score_cluster['metrics']['f1_score']*score_cluster['size']
                 total_size += score_cluster['size']

             average_score = average_score / total_size
             print(f"average f1 on clusters {100*average_score:0.1f}% gain {100*(average_score-r
```

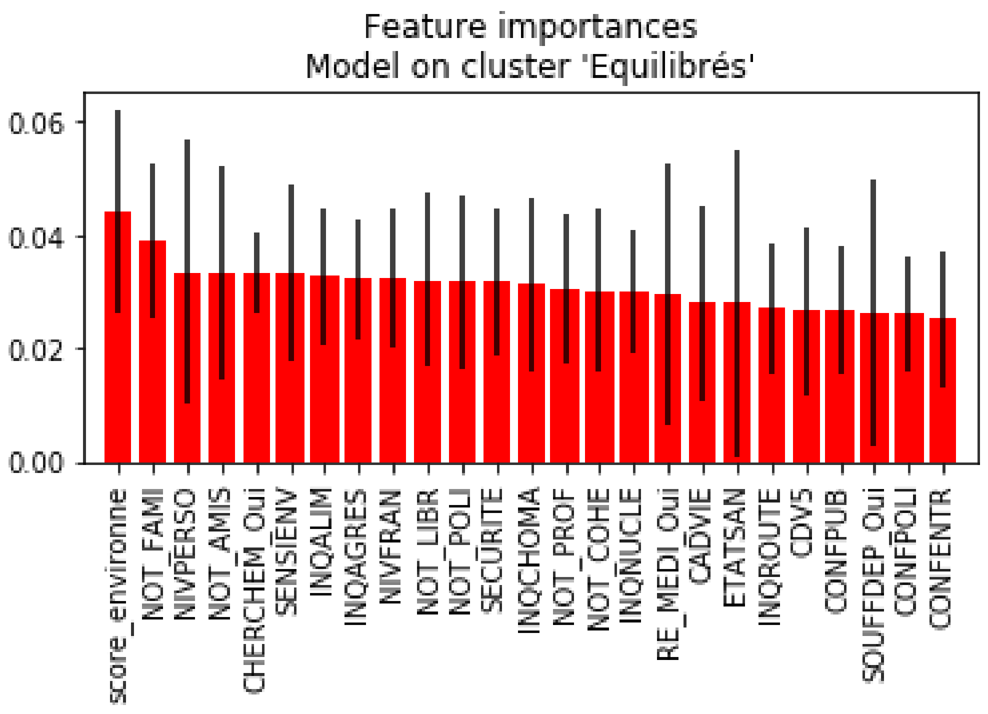
```
method clust3:
cluster 2 'Equilibrés' : (3053), f1 macro 85.6%
cluster 4 'Domestiques modérés' : (2359), f1 macro 90.5%
cluster 6 'Domestiques stricts' : (2313), f1 macro 90.4%
cluster 1 'Civiques' : (528), f1 macro 92.1%
cluster 3 'Flâneurs' : (1384), f1 macro 95.4%
cluster 5 'Solitaires' : (1494), f1 macro 87.8%
average f1 on clusters 89.5% gain 9.5
```

```
In [38]: # accuracy
         for score_method in score_clustering_methods:
             print(f"method {score_method['clustering_method']}:")
             average_score = 0
             total_size = 0
             for i, score_cluster in enumerate(score_method['cluster_scores']):
                 print(f"cluster {score_cluster['cluster']} '{cluster_name[score_cluster['cluste
                 average_score = average_score + score_cluster['metrics']['accuracy']*score_clus
                 total_size += score_cluster['size']
             average_score = average_score / total_size
             print(f"average accuracy on clusters {100*average_score:0.1f}% gain {100*(average_s
```

```
method clust3:
cluster 2 'Equilibrés' : (3053) accuracy 81.5%
cluster 4 'Domestiques modérés' : (2359) accuracy 86.8%
cluster 6 'Domestiques stricts' : (2313) accuracy 86.2%
cluster 1 'Civiques' : (528) accuracy 92.6%
cluster 3 'Flâneurs' : (1384) accuracy 93.3%
cluster 5 'Solitaires' : (1494) accuracy 84.5%
average accuracy on clusters 86.0% gain 13.0
```

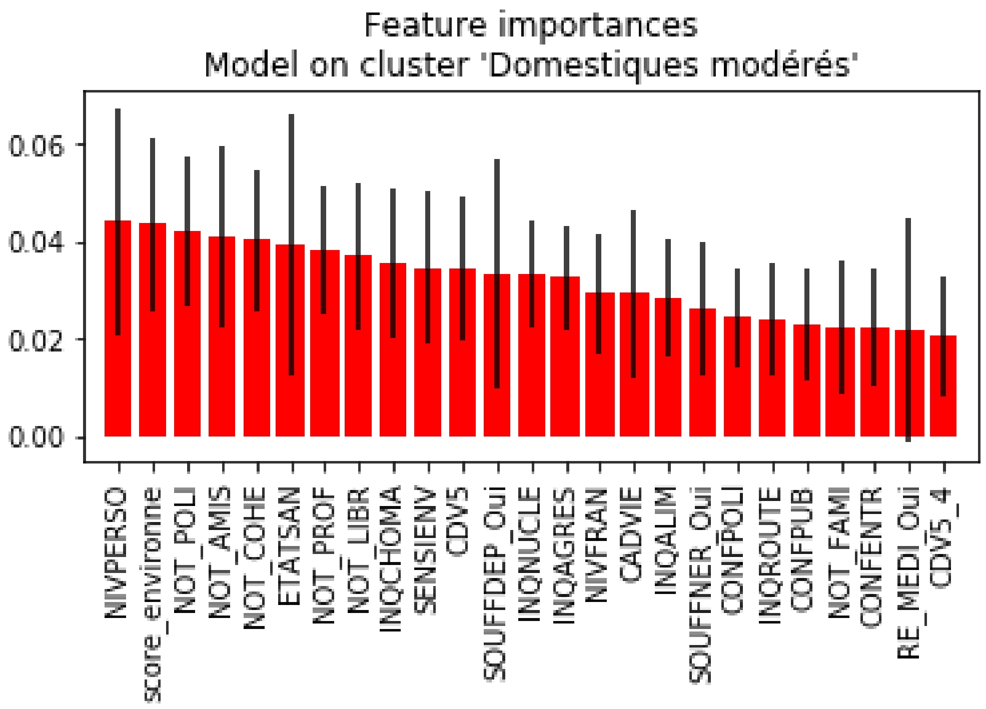
```
In [39]: # Feature importance by cluster
         for score_method in score_clustering_methods:
             print(f"method {score_method['clustering_method']}:")
             for i, score_cluster in enumerate(score_method['cluster_scores']):
                 print(f"cluster {score_cluster['cluster']} ({score_cluster['size']}), f1 macro
                 print(f"top 15 features:")
                 indices = score_cluster['indices']
                 features_name = score_cluster['features_name']
                 importances = score_cluster['importances']
                 features_short_name_sorted = [ name[:15] for name in features_name[indices]]

                 # Plot the feature importances of the forest
                 plt.figure()
                 plt.gcf().subplots_adjust(bottom=0.4)
                 plt.title(f"Feature importances\nModel on cluster '{cluster_name[score_cluster[
                 plt.bar(range(n_features), importances[indices][:n_features],
                         color="r", yerr=std[indices][:n_features], align="center")
                 plt.xticks(range(n_features), features_short_name_sorted[:n_features], rotation
                 plt.xlim([-1, n_features])
                 filename = path_fig / Path(f"feature_importance_cluster_{score_cluster['cluster
                 plt.savefig(filename, format='jpg')
                 plt.show()

                 for f in range(n_features_max):
                     print("%d. feature %d -%s- (%f)" % (f + 1, indices[f],features_name[indices
                     if features_name[indices[f]] in actionable_individual_1_features:
                         print("\tActionable at individual level (1)")
                     if features_name[indices[f]] in actionable_individual_2_features:
                         print("\tActionable at individual level (2)")
                     if features_name[indices[f]] in actionable_admin_1_features:
                         print("\tActionable at administrative level (1)")
                     if features_name[indices[f]] in actionable_admin_2_features:
                         print("\tActionable at administrative level (2)")
```

```
method clust3:
cluster 2 (3053), f1 macro 85.6%
top 15 features:
```



```
1. feature 28 -score_environnement- (0.044090)
2. feature 14 -NOT_FAMI- (0.038867)
     Actionable at individual level (1)
     Actionable at administrative level (2)
3. feature 27 -NIVPERSO- (0.033454)
     Actionable at individual level (2)
     Actionable at administrative level (2)
4. feature 36 -NOT_AMIS- (0.033307)
     Actionable at individual level (1)
     Actionable at administrative level (2)
5. feature 22 -CHERCHEM_Oui- (0.033212)
     Actionable at individual level (1)
     Actionable at administrative level (2)
6. feature 32 -SENSIENV- (0.033058)
     Actionable at individual level (1)
     Actionable at administrative level (1)
7. feature 2 -INQALIM- (0.032553)
     Actionable at individual level (1)
     Actionable at administrative level (1)
8. feature 9 -INQAGRES- (0.032290)
     Actionable at individual level (1)
     Actionable at administrative level (1)
9. feature 8 -NIVFRAN- (0.032261)
     Actionable at individual level (2)
```

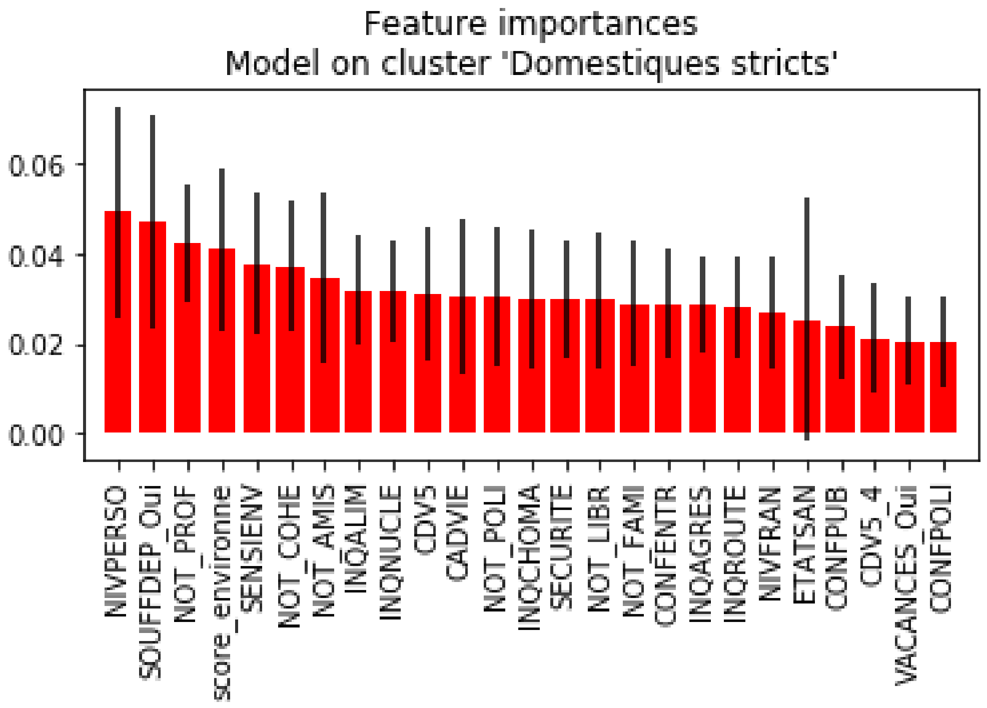
```
        Actionable at administrative level (1)
10. feature 24 -NOT_LIBR- (0.031972)
        Actionable at individual level (1)
        Actionable at administrative level (1)
11. feature 33 -NOT_POLI- (0.031706)
        Actionable at individual level (1)
        Actionable at administrative level (1)
12. feature 17 -SECURITE- (0.031628)
        Actionable at individual level (2)
        Actionable at administrative level (1)
13. feature 20 -INQCHOMA- (0.031245)
        Actionable at individual level (1)
        Actionable at administrative level (1)
14. feature 26 -NOT_PROF- (0.030478)
        Actionable at individual level (1)
        Actionable at administrative level (2)
15. feature 31 -NOT_COHE- (0.030046)
        Actionable at individual level (1)
        Actionable at administrative level (1)
16. feature 39 -INQNUCLE- (0.029950)
        Actionable at individual level (1)
        Actionable at administrative level (1)
17. feature 29 -RE_MEDI_Oui- (0.029334)
        Actionable at individual level (2)
        Actionable at administrative level (2)
18. feature 16 -CADVIE- (0.027998)
        Actionable at individual level (1)
        Actionable at administrative level (1)
19. feature 3 -ETATSAN- (0.027841)
        Actionable at individual level (1)
        Actionable at administrative level (1)
20. feature 11 -INQROUTE- (0.026949)
        Actionable at individual level (1)
        Actionable at administrative level (1)
21. feature 7 -CDV5- (0.026573)
        Actionable at individual level (2)
        Actionable at administrative level (1)
22. feature 6 -CONFPUB- (0.026497)
        Actionable at individual level (1)
        Actionable at administrative level (1)
23. feature 35 -SOUFFDEP_Oui- (0.026268)
        Actionable at individual level (2)
        Actionable at administrative level (1)
24. feature 30 -CONFPOLI- (0.025985)
        Actionable at individual level (1)
        Actionable at administrative level (1)
25. feature 21 -CONFENTR- (0.025161)
        Actionable at individual level (1)
        Actionable at administrative level (1)
cluster 4 (2359), f1 macro 90.5%
top 15 features:
```



```
1. feature 27 -NIVPERSO- (0.043900)
        Actionable at individual level (2)
        Actionable at administrative level (2)
2. feature 28 -score_environnement- (0.043421)
3. feature 33 -NOT_POLI- (0.041819)
        Actionable at individual level (1)
        Actionable at administrative level (1)
4. feature 36 -NOT_AMIS- (0.040947)
        Actionable at individual level (1)
        Actionable at administrative level (2)
5. feature 31 -NOT_COHE- (0.040179)
        Actionable at individual level (1)
        Actionable at administrative level (1)
6. feature 3 -ETATSAN- (0.039159)
        Actionable at individual level (1)
        Actionable at administrative level (1)
7. feature 26 -NOT_PROF- (0.038041)
        Actionable at individual level (1)
        Actionable at administrative level (2)
```

```
8. feature 24 -NOT_LIBR- (0.036783)
        Actionable at individual level (1)
        Actionable at administrative level (1)
9. feature 20 -INQCHOMA- (0.035285)
        Actionable at individual level (1)
        Actionable at administrative level (1)
10. feature 32 -SENSIENV- (0.034527)
        Actionable at individual level (1)
        Actionable at administrative level (1)
11. feature 7 -CDV5- (0.034380)
        Actionable at individual level (2)
        Actionable at administrative level (1)
12. feature 35 -SOUFFDEP_Oui- (0.033361)
        Actionable at individual level (2)
        Actionable at administrative level (1)
13. feature 39 -INQNUCLE- (0.033274)
        Actionable at individual level (1)
        Actionable at administrative level (1)
14. feature 9 -INQAGRES- (0.032460)
        Actionable at individual level (1)
        Actionable at administrative level (1)
15. feature 8 -NIVFRAN- (0.029314)
        Actionable at individual level (2)
        Actionable at administrative level (1)
16. feature 16 -CADVIE- (0.029248)
        Actionable at individual level (1)
        Actionable at administrative level (1)
17. feature 2 -INQALIM- (0.028259)
        Actionable at individual level (1)
        Actionable at administrative level (1)
18. feature 38 -SOUFFNER_Oui- (0.026139)
        Actionable at individual level (2)
        Actionable at administrative level (1)
19. feature 30 -CONFPOLI- (0.024280)
        Actionable at individual level (1)
        Actionable at administrative level (1)
20. feature 11 -INQROUTE- (0.023849)
        Actionable at individual level (1)
        Actionable at administrative level (1)
21. feature 6 -CONFPUB- (0.022800)
        Actionable at individual level (1)
        Actionable at administrative level (1)
22. feature 14 -NOT_FAMI- (0.022355)
        Actionable at individual level (1)
        Actionable at administrative level (2)
23. feature 21 -CONFENTR- (0.021992)
        Actionable at individual level (1)
        Actionable at administrative level (1)
```

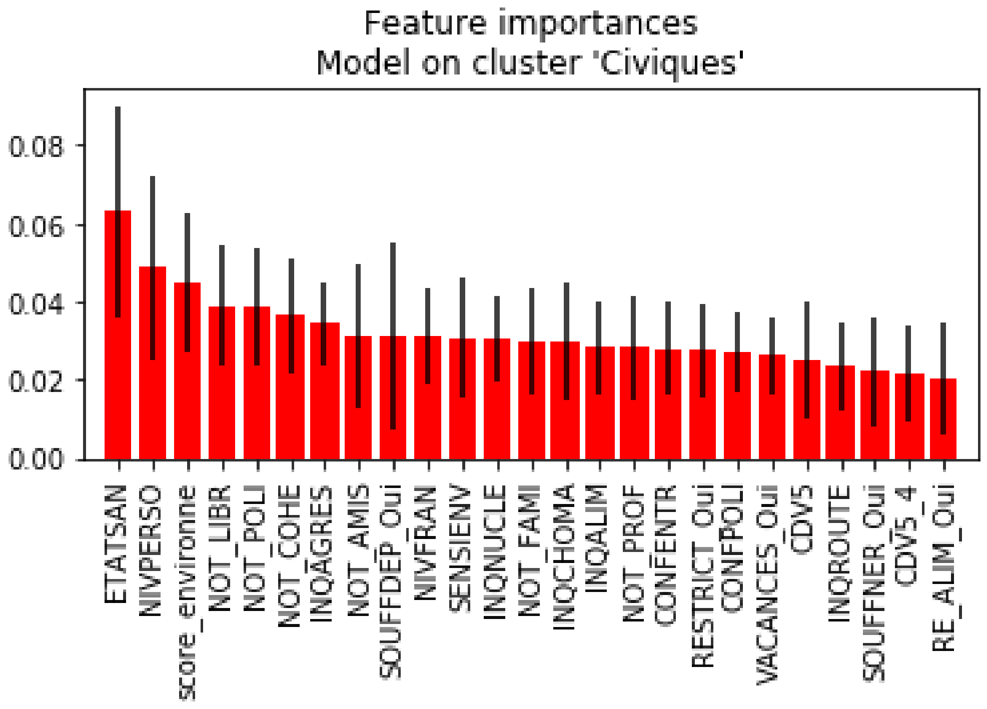
```
24. feature 29 -RE_MEDI_Oui- (0.021655)
        Actionable at individual level (2)
        Actionable at administrative level (2)
25. feature 4 -CDV5_4- (0.020346)
        Actionable at individual level (2)
        Actionable at administrative level (1)
cluster 6 (2313), f1 macro 90.4%
top 15 features:
```



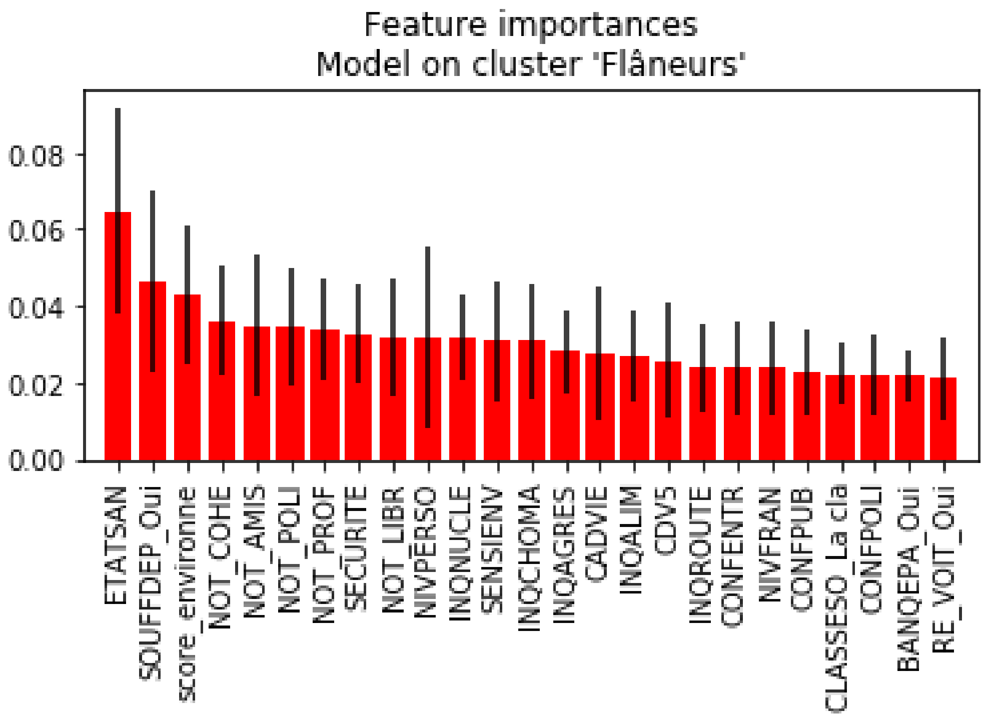
```
1. feature 27 -NIVPERSO- (0.048936)
        Actionable at individual level (2)
        Actionable at administrative level (2)
2. feature 35 -SOUFFDEP_Oui- (0.046825)
        Actionable at individual level (2)
        Actionable at administrative level (1)
3. feature 26 -NOT_PROF- (0.042092)
        Actionable at individual level (1)
        Actionable at administrative level (2)
4. feature 28 -score_environnement- (0.040757)
5. feature 32 -SENSIENV- (0.037539)
        Actionable at individual level (1)
        Actionable at administrative level (1)
6. feature 31 -NOT_COHE- (0.036953)
```

```
        Actionable at individual level (1)
        Actionable at administrative level (1)
7. feature 36 -NOT_AMIS- (0.034509)
        Actionable at individual level (1)
        Actionable at administrative level (2)
8. feature 2 -INQALIM- (0.031748)
        Actionable at individual level (1)
        Actionable at administrative level (1)
9. feature 39 -INQNUCLE- (0.031542)
        Actionable at individual level (1)
        Actionable at administrative level (1)
10. feature 7 -CDV5- (0.031029)
        Actionable at individual level (2)
        Actionable at administrative level (1)
11. feature 16 -CADVIE- (0.030458)
        Actionable at individual level (1)
        Actionable at administrative level (1)
12. feature 33 -NOT_POLI- (0.030329)
        Actionable at individual level (1)
        Actionable at administrative level (1)
13. feature 20 -INQCHOMA- (0.029968)
        Actionable at individual level (1)
        Actionable at administrative level (1)
14. feature 17 -SECURITE- (0.029904)
        Actionable at individual level (2)
        Actionable at administrative level (1)
15. feature 24 -NOT_LIBR- (0.029460)
        Actionable at individual level (1)
        Actionable at administrative level (1)
16. feature 14 -NOT_FAMI- (0.028673)
        Actionable at individual level (1)
        Actionable at administrative level (2)
17. feature 21 -CONFENTR- (0.028642)
        Actionable at individual level (1)
        Actionable at administrative level (1)
18. feature 9 -INQAGRES- (0.028567)
        Actionable at individual level (1)
        Actionable at administrative level (1)
19. feature 11 -INQROUTE- (0.027896)
        Actionable at individual level (1)
        Actionable at administrative level (1)
20. feature 8 -NIVFRAN- (0.026827)
        Actionable at individual level (2)
        Actionable at administrative level (1)
21. feature 3 -ETATSAN- (0.025247)
        Actionable at individual level (1)
        Actionable at administrative level (1)
22. feature 6 -CONFPUB- (0.023631)
```

```
        Actionable at individual level (1)
        Actionable at administrative level (1)
23. feature 4 -CDV5_4- (0.021140)
        Actionable at individual level (2)
        Actionable at administrative level (1)
24. feature 25 -VACANCES_Oui- (0.020555)
        Actionable at individual level (1)
        Actionable at administrative level (1)
25. feature 30 -CONFPOLI- (0.020254)
        Actionable at individual level (1)
        Actionable at administrative level (1)
cluster 1 (528), f1 macro 92.1%
top 15 features:
```



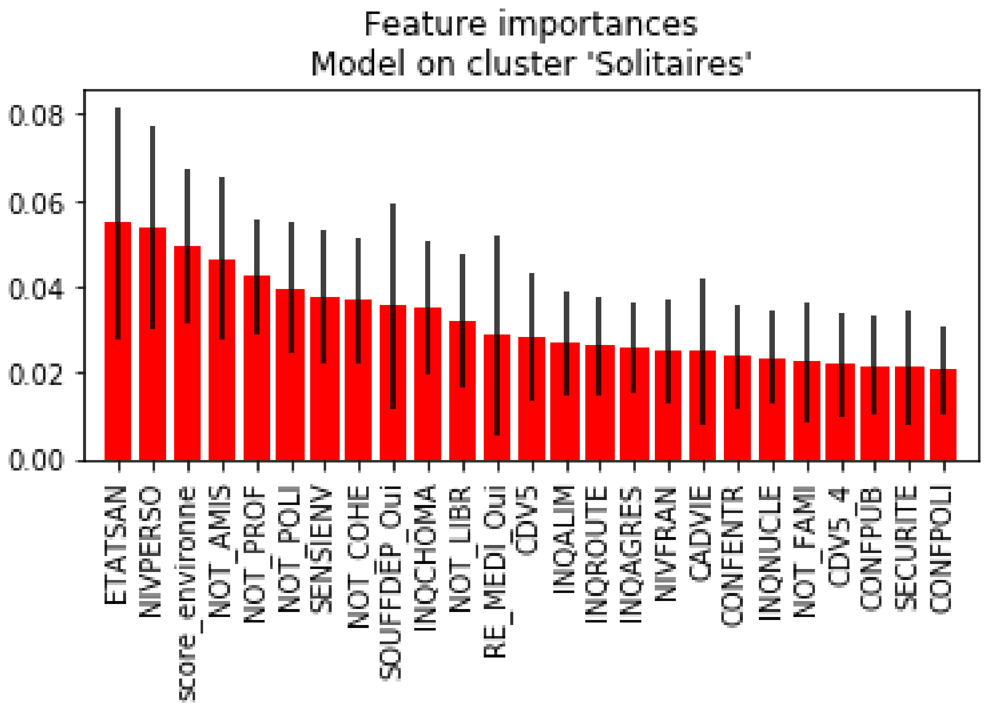
```
1. feature 3 -ETATSAN- (0.063048)
        Actionable at individual level (1)
        Actionable at administrative level (1)
2. feature 27 -NIVPERSO- (0.048830)
        Actionable at individual level (2)
        Actionable at administrative level (2)
3. feature 28 -score_environnement- (0.044799)
4. feature 24 -NOT_LIBR- (0.038897)
        Actionable at individual level (1)
        Actionable at administrative level (1)
5. feature 33 -NOT_POLI- (0.038633)
        Actionable at individual level (1)
        Actionable at administrative level (1)
6. feature 31 -NOT_COHE- (0.036256)
        Actionable at individual level (1)
        Actionable at administrative level (1)
7. feature 9 -INQAGRES- (0.034210)
        Actionable at individual level (1)
        Actionable at administrative level (1)
8. feature 36 -NOT_AMIS- (0.031163)
        Actionable at individual level (1)
        Actionable at administrative level (2)
9. feature 35 -SOUFFDEP_Oui- (0.031066)
        Actionable at individual level (2)
        Actionable at administrative level (1)
10. feature 8 -NIVFRAN- (0.030870)
        Actionable at individual level (2)
        Actionable at administrative level (1)
11. feature 32 -SENSIENV- (0.030765)
        Actionable at individual level (1)
        Actionable at administrative level (1)
12. feature 39 -INQNUCLE- (0.030399)
        Actionable at individual level (1)
        Actionable at administrative level (1)
13. feature 14 -NOT_FAMI- (0.029980)
        Actionable at individual level (1)
        Actionable at administrative level (2)
14. feature 20 -INQCHOMA- (0.029647)
        Actionable at individual level (1)
        Actionable at administrative level (1)
15. feature 2 -INQALIM- (0.028161)
        Actionable at individual level (1)
        Actionable at administrative level (1)
16. feature 26 -NOT_PROF- (0.028152)
        Actionable at individual level (1)
        Actionable at administrative level (2)
17. feature 21 -CONFENTR- (0.027804)
        Actionable at individual level (1)
        Actionable at administrative level (1)
18. feature 23 -RESTRICT_Oui- (0.027385)
        Actionable at individual level (2)
        Actionable at administrative level (2)
19. feature 30 -CONFPOLI- (0.027211)
        Actionable at individual level (1)
        Actionable at administrative level (1)
20. feature 25 -VACANCES_Oui- (0.026125)
        Actionable at individual level (1)
        Actionable at administrative level (1)
21. feature 7 -CDV5- (0.025027)
        Actionable at individual level (2)
        Actionable at administrative level (1)
22. feature 11 -INQROUTE- (0.023451)
        Actionable at individual level (1)
        Actionable at administrative level (1)
23. feature 38 -SOUFFNER_Oui- (0.021919)
        Actionable at individual level (2)
        Actionable at administrative level (1)
24. feature 4 -CDV5_4- (0.021575)
        Actionable at individual level (2)
        Actionable at administrative level (1)
25. feature 1 -RE_ALIM_Oui- (0.020150)
        Actionable at individual level (2)
        Actionable at administrative level (2)
cluster 3 (1384), f1 macro 95.4%
top 15 features:
```



```
1. feature 3 -ETATSAN- (0.064748)
        Actionable at individual level (1)
        Actionable at administrative level (1)
2. feature 35 -SOUFFDEP_Oui- (0.046520)
```

```
        Actionable at individual level (2)
        Actionable at administrative level (1)
3. feature 28 -score_environnement- (0.042737)
4. feature 31 -NOT_COHE- (0.036064)
        Actionable at individual level (1)
        Actionable at administrative level (1)
5. feature 36 -NOT_AMIS- (0.034809)
        Actionable at individual level (1)
        Actionable at administrative level (2)
6. feature 33 -NOT_POLI- (0.034676)
        Actionable at individual level (1)
        Actionable at administrative level (1)
7. feature 26 -NOT_PROF- (0.033644)
        Actionable at individual level (1)
        Actionable at administrative level (2)
8. feature 17 -SECURITE- (0.032646)
        Actionable at individual level (2)
        Actionable at administrative level (1)
9. feature 24 -NOT_LIBR- (0.031848)
        Actionable at individual level (1)
        Actionable at administrative level (1)
10. feature 27 -NIVPERSO- (0.031829)
        Actionable at individual level (2)
        Actionable at administrative level (2)
11. feature 39 -INQNUCLE- (0.031628)
        Actionable at individual level (1)
        Actionable at administrative level (1)
12. feature 32 -SENSIENV- (0.030718)
        Actionable at individual level (1)
        Actionable at administrative level (1)
13. feature 20 -INQCHOMA- (0.030694)
        Actionable at individual level (1)
        Actionable at administrative level (1)
14. feature 9 -INQAGRES- (0.027941)
        Actionable at individual level (1)
        Actionable at administrative level (1)
15. feature 16 -CADVIE- (0.027490)
        Actionable at individual level (1)
        Actionable at administrative level (1)
16. feature 2 -INQALIM- (0.027016)
        Actionable at individual level (1)
        Actionable at administrative level (1)
17. feature 7 -CDV5- (0.025719)
        Actionable at individual level (2)
        Actionable at administrative level (1)
18. feature 11 -INQROUTE- (0.023945)
        Actionable at individual level (1)
        Actionable at administrative level (1)
```

```
19. feature 21 -CONFENTR- (0.023823)
        Actionable at individual level (1)
        Actionable at administrative level (1)
20. feature 8 -NIVFRAN- (0.023807)
        Actionable at individual level (2)
        Actionable at administrative level (1)
21. feature 6 -CONFPUB- (0.022498)
        Actionable at individual level (1)
        Actionable at administrative level (1)
22. feature 15 -CLASSESO_La classe moyenne supérieure- (0.022294)
        Actionable at individual level (1)
        Actionable at administrative level (1)
23. feature 30 -CONFPOLI- (0.021941)
        Actionable at individual level (1)
        Actionable at administrative level (1)
24. feature 18 -BANQEPA_Oui- (0.021660)
        Actionable at individual level (1)
        Actionable at administrative level (1)
25. feature 5 -RE_VOIT_Oui- (0.020998)
        Actionable at individual level (2)
        Actionable at administrative level (2)
cluster 5 (1494), f1 macro 87.8%
top 15 features:
```



```
1. feature 3 -ETATSAN- (0.054595)
        Actionable at individual level (1)
        Actionable at administrative level (1)
2. feature 27 -NIVPERSO- (0.053860)
        Actionable at individual level (2)
        Actionable at administrative level (2)
3. feature 28 -score_environnement- (0.048987)
4. feature 36 -NOT_AMIS- (0.046352)
        Actionable at individual level (1)
        Actionable at administrative level (2)
5. feature 26 -NOT_PROF- (0.042180)
        Actionable at individual level (1)
        Actionable at administrative level (2)
6. feature 33 -NOT_POLI- (0.039579)
        Actionable at individual level (1)
        Actionable at administrative level (1)
7. feature 32 -SENSIENV- (0.037516)
        Actionable at individual level (1)
        Actionable at administrative level (1)
8. feature 31 -NOT_COHE- (0.036678)
        Actionable at individual level (1)
        Actionable at administrative level (1)
9. feature 35 -SOUFFDEP_Oui- (0.035340)
        Actionable at individual level (2)
        Actionable at administrative level (1)
10. feature 20 -INQCHOMA- (0.034935)
        Actionable at individual level (1)
        Actionable at administrative level (1)
11. feature 24 -NOT_LIBR- (0.031904)
        Actionable at individual level (1)
        Actionable at administrative level (1)
12. feature 29 -RE_MEDI_Oui- (0.028559)
        Actionable at individual level (2)
        Actionable at administrative level (2)
13. feature 7 -CDV5- (0.028030)
        Actionable at individual level (2)
        Actionable at administrative level (1)
14. feature 2 -INQALIM- (0.026677)
        Actionable at individual level (1)
        Actionable at administrative level (1)
15. feature 11 -INQROUTE- (0.026259)
        Actionable at individual level (1)
        Actionable at administrative level (1)
16. feature 9 -INQAGRES- (0.025731)
        Actionable at individual level (1)
        Actionable at administrative level (1)
17. feature 8 -NIVFRAN- (0.024799)
        Actionable at individual level (2)
        Actionable at administrative level (1)
18. feature 16 -CADVIE- (0.024798)
        Actionable at individual level (1)
        Actionable at administrative level (1)
19. feature 21 -CONFENTR- (0.023701)
        Actionable at individual level (1)
        Actionable at administrative level (1)
20. feature 39 -INQNUCLE- (0.023500)
        Actionable at individual level (1)
        Actionable at administrative level (1)
21. feature 14 -NOT_FAMI- (0.022355)
        Actionable at individual level (1)
        Actionable at administrative level (2)
22. feature 4 -CDV5_4- (0.021720)
        Actionable at individual level (2)
        Actionable at administrative level (1)
23. feature 6 -CONFPUB- (0.021568)
        Actionable at individual level (1)
        Actionable at administrative level (1)
24. feature 17 -SECURITE- (0.021161)
        Actionable at individual level (2)
        Actionable at administrative level (1)
25. feature 30 -CONFPOLI- (0.020678)
        Actionable at individual level (1)
        Actionable at administrative level (1)
```