Felix_dataset_preparation

October 21, 2018

1 Felix- dataset preparation

A dataset is build based on "Etude condistions de vie" merged with "INSEE" communal data ... . Categorial features are ... K-1 categ ... Set of variables and featiures ...

```
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 pickle
        #%pylab inline

        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import StandardScaler
        from sklearn.linear_model import LogisticRegression
        from sklearn.model_selection import cross_val_score, GridSearchCV
        from sklearn.decomposition import PCA
        from sklearn.ensemble import RandomForestClassifier
        from sklearn.metrics import confusion_matrix, f1_score, precision_score, recall_score
        from sklearn.preprocessing import LabelEncoder
        from sklearn.preprocessing import LabelBinarizer
        from sklearn.preprocessing import OneHotEncoder
        from sklearn.svm import SVC, LinearSVC
        from sklearn.model_selection import StratifiedKFold
        from sklearn.feature_selection import RFECV, RFE, SelectKBest, chi2, SelectFromModel
        from sklearn.utils import resample
```

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

1.1 III - Feature sets and engineering - Dataset preparation

```
In [3]: # loading cdv data
        file = path_data / Path("felix.csv")
        with Path.open(file, 'rb') as fp:
            cdv = pd.read_csv(fp,  encoding='cp1252',low_memory=False, index_col = 0)
```

```
In [4]: # loadind cdv data without format
        file = path_data / Path("felix_ssfmt.csv")
        with Path.open(file, 'rb') as fp:
            cdv_ssfmt = pd.read_csv(fp,  encoding='cp1252',low_memory=False, index_col = 0)
```

1.1.1 Feature scope

```
In [5]: # number of line per year in teh dataset
        n_per_year = cdv["ANNEEFUZ"].value_counts()
        # number of missing value per variable for a given year
        na_2015 = np.sum(cdv.loc[cdv["ANNEEFUZ"] == 2015].isnull())
        na_2016 = np.sum(cdv.loc[cdv["ANNEEFUZ"] == 2016].isnull())
        na_2017 = np.sum(cdv.loc[cdv["ANNEEFUZ"] == 2017].isnull())
        na_2018 = np.sum(cdv.loc[cdv["ANNEEFUZ"] == 2018].isnull())
        # column scope per year
        cdv_2015_var = set(na_2015[na_2015 < n_per_year[2015]].index)
        cdv_2016_var = set(na_2016[na_2016 < n_per_year[2016]].index)
        cdv_2017_var = set(na_2017[na_2017 < n_per_year[2017]].index)
        cdv_2018_var = set(na_2018[na_2018 < n_per_year[2018]].index)
```

```
In [6]: cdv_2015_2018_var = (cdv_2015_var & cdv_2016_var & cdv_2017_var & cdv_2018_var)
        cdv_2016_2018_var = (cdv_2016_var & cdv_2017_var & cdv_2018_var)
        cdv_2017_2018_var = (cdv_2017_var & cdv_2018_var)
```

```
In [7]: print(f"{len(cdv_2015_2018_var)} variables common to all study out of {cdv_ssfmt.shape[1
```

```
267 variables common to all study out of 353
```

1.1.2 Special variables

```
In [8]: pred_var = {"HEUREUX"}
        tech_var = {"ANNEEFUZ", "ANNEFUZ2", "COLLECTE", "CHAMP",
                    "identifiant", "an_enq", "INTER"}
        com_var = {'COMINSEE', 'DEPCOM', 'com', 'inseel','inseenum','CP'}
        text_var = {'RADIQUOI'}
        bizz_var = {'NB0003','NB0306','NB0610','NB1016','NB1620','NB2099',
                    'an_nais','decuc','decsqt','info','typodeg','refus2',
                    'cpt', 'prescaf', 'poptrpeu','REVUC','i','REVTOT',
                    'poppeud','popdense', 'popinter', 'pmun', 'agedip', 'age_OW',
                    'REVsqt', 'NBUC', 'AGGLOINS', 'med', 'CSP6','REVTOT6',
                    'ACM1','ACM2','ACM3','ACM4','ACM5', 'ACM6', 'ACM7',
                    'ACM8','ACM9', 'ACM10', 'ACM11','ACM12'}
```

1.1.3 Categorial variable

```
In [9]: obj_cdv = cdv.select_dtypes(include=['object'])
        obj_var = set(obj_cdv.columns)
        cat_max9_var = set()
        cat_min10_var = set()
        for c in obj_var:
            obj_cdv_valcpt = obj_cdv[c].value_counts()
            if len(obj_cdv_valcpt) > 10:
                cat_min10_var.add(c)
            else:
                cat_max9_var.add(c)
```

```
In [10]: ord_var = {
             "CONFPOLI", "AGE5","SECURITE", "ACM7","INQGUERR", "NBPIECE6","INNOVTEC",
             "JUSTICE","EFFORTPP","ACM10","NIVFRAN4","NBPERS5","INQCHOMA","CDV5_4",
             "CONFGOUV","ADOPTGAY","ACM8","FREQCINE","CONFPUB","FREQSPOR","INQALIM",
             "ASSO10_3", "FREQBIBL","DEPLOG","NBCHOM","CONFENTR","ORDLIB","ACM5",
             "INQMALAD","FREQTELE","NBENF6","ACM9","revtot7","INQROUTE","NIVPERS4",
             "ETATSAN","INQNUCLE","NIVPERSO","CONFASSO","ACM6","CDV5","UNIONGAY",
             "ACM4","INQAGRES","CADVIE","NIVFRAN","REV_TR7","ISEGO","RECEP","AGE6",
             "ADNCB","PRATCOLL","NBHEUR39","HARVEY","QUOTAAGE","NBHEUR35","RELEG",
             "CONFKEUF","CONFECOL","ADNSTIC","ADNORDI","CONFPRES","CONFWEB","CONFBANK"
         }
```

```
In [11]: # exclusion of features with order
         cat_var = obj_var - ord_var
         cat_max9_var = cat_max9_var - ord_var
         cat_min10_var = cat_min10_var - ord_var
```

```
In [12]: cdv_dtypes = cdv.dtypes
```

```
In [13]: int_var = set(cdv_dtypes[cdv_dtypes == 'int64'].index)
```

```
In [14]: int_cat_var = {
             'NB0003','NB0306','NB0610','NB1016',
             'NB1620','NB2099', 'REVTOT6','ANNEEFUZ','INTER', 'INTER6'
         }
         int_cat_max9_var = {
             'NB0003','NB0306','NB0610','NB1016',
             'NB1620','NB2099', "REVTOT6","ANNEEFUZ"}
         int_cat_min10_var = {
             'INTER', 'INTER6'
         }
         int_quant_var = {
             'AGE','REVENQ', 'AUTREREV',
             'an_enq','an_nais'
         }
```

```
In [15]: cat_var = cat_var | int_cat_var
         cat_max9_var = cat_max9_var | int_cat_max9_var
         cat_min10_var = cat_min10_var | int_cat_min10_var
         quant_var = ord_var | int_quant_var
```

```
In [16]: float_var = set(cdv_dtypes[cdv_dtypes == 'float64'].index)
```

```
In [17]: float_cat_min10_var = {'CP','inseenum'}
         float_cat_max9_var = {'refus2','cpt','prescaf','i','age_OW','TYPLOG','AGGLOINS','CSP6'}
         float_cat_var = float_cat_min10_var | float_cat_max9_var
         float_quant_var = float_var - float_cat_var
```

```
In [18]: cat_var = cat_var | float_cat_var
         cat_max9_var = cat_max9_var | float_cat_max9_var
         cat_min10_var = cat_min10_var | float_cat_min10_var
         quant_var = quant_var | float_quant_var
```

```
In [19]: print(f"out of the {cdv.shape[1]} variable :")
         print(f"{len(cat_var)} variables are categorial ")
         print(f"{len(quant_var)} variables are quantitative ")
```

```
out of the 353 variable :
248 variables are categorial
106 variables are quantitative
```

```
In [20]: print(f"out of the {len(cat_var)} variable categorial:")
         print(f"{len(cat_max9_var)} variables have maximum 9 modalities  ")
         print(f"{len(cat_min10_var)} variables have more ")
```

```
out of the 248 variable categorial:
221 variables have maximum 9 modalities
27 variables have more
```

```
In [21]: dict_var_groups = {
             'cdv_2015_var' : cdv_2015_var,
             'cdv_2016_var' : cdv_2016_var,
             'cdv_2017_var' : cdv_2017_var,
             'cdv_2018_var' : cdv_2018_var,
             'cdv_2015_2018_var' : cdv_2015_2018_var,
             'cdv_2016_2018_var' : cdv_2016_2018_var,
             'cdv_2017_2018_var' : cdv_2017_2018_var,
             'pred_var' : pred_var,
             'tech_var' : tech_var,
             'com_var' : com_var,
             'text_var' : text_var,
             'bizz_var' : bizz_var,
             'cat_var' : cat_var,
             'cat_max9_var' : cat_max9_var,
             'cat_min10_var' : cat_min10_var,
             'quant_var' : quant_var
         }
```

1.1.4 Adding communal features and levers

```
In [22]: # loading MergeCommunesEnvi data
         file = path_data / Path("MergeCommunesEnvi.csv")
         with Path.open(file, 'rb') as fp:
             MergeCommunesEnvi = pd.read_csv(fp,  encoding='cp1252',low_memory=False, sep=';', i
```

```
In [23]: MergeCommunesEnvi.shape
```

```
Out[23]: (11131, 571)
```

```
In [24]: # file 'List-of-Actionable-Variables_v0.1_sp' september 01
         indiv_act_var = {
             "LIMVIAND","VACANCES","VISITFAM","RECEP","YOGA","FREQSPOR","FREQBIBL","FREQCINE",
             "FREQTELE","ASSOSPOR","ASSOCULT","ASSOCONF","ASSOJEUN","ASSOSYND","ASSOENVI",
             "ASSOPARE","ASSOCONS","ASSOPOLI","ASSOHUMA","ASSOAUTR","NOT_FAMI","NOT_PROF",
             "NOT_AMIS","NOT_COHE","NOT_POLI","NOT_LIBR","NOT_LOG","NOT_CAD","RELIGION"
         }
```

```
In [25]: # file 'List-of-Actionable-Variables_v0.1_sp' september 01
         indiv_semi_act_var = {
             "SITUEMP5","SITUEMP6","TEMPSTRA","nbheures","NBHEUR39","NBHEUR35",
             "IMAGTRAV","COUPLE","ENFANTS","CADVIE","CADVIE3","MODCHAUF","ETATSAN",
             "BANQMOB","BANQEPA","BANQVIE","TELMOB","CONFPUB","CONFENTR","CONFASSO",
             "CONFPOLI","CONFBANK","CONFPRES","CONFECOL","CONFKEUF","INQMALAD",
             "INQMALA3","INQAGRES","INQAGRE3","INQROUTE","INQROUT3","INQCHOMA",
             "INQCHOM3","INQGUERR","INQGUER3","INQNUCLE","INQNUCL3","INQALIM",
             "INQALIM3","ECHPOL"
         }
```

```
In [26]: admin_act_var = {
             "AIDESUFF","EFFORTPP","CHOAVANT","OPIRSA","JUSTICE","RELEG","RADIQUOI",
             "RADWHY1","RADWHY2","RADWHY3","RADWHY4","RADWHY5","RADWHY6","RADWHY7",
             "RADWHY8","RADWHY9","RADWHY10","RADWHY11","RADWHY12","RADWHY13","RADWHY14",
             "ORDLIB","PREOCCU1","PREOCCU2","CONFGOUV"
         }
```

```
In [27]: admin_semi_act_var = {
             "SECURITE","SECUR3","ADNSTIC","ADNCB","ADNORDI","ROBOT1","ROBOT2","ROBOT3",
             "PRESTCAF","REVPF","CONFPUB","CONFENTR","CONFASSO","CONFPOLI","CONFBANK",
             "CONFPRES","CONFECOL","CONFKEUF","TRANSFST","TRANSFO5","PROGRAD","OPIIMMIG"
         }
```

```
In [28]: commune_var = set(MergeCommunesEnvi.columns) - set(cdv.columns)
In [29]: commune_var

Out[29]: {'DECE1015',
          'DECESD16',
          'DEP',
          'ETAZ15',
          'ETBE15',
          'ETFZ15',
          'ETGU15',
          'ETGZ15',
          'ETOQ15',
          'ETTEF115',
          'ETTEFP1015',
          'ETTOT15',
          'LIBGEO',
          'MED15',
          'NAIS1015',
          'NAISD16',
          'NBMENFISC15',
          'NB_A101',
          'NB_A104',
          'NB_A105',
          'NB_A106',
          'NB_A107',
          'NB_A108',
          'NB_A109',
          'NB_A115',
          'NB_A119',
          'NB_A120',
          'NB_A121',
          'NB_A122',
          'NB_A123',
          'NB_A124',
          'NB_A125',
          'NB_A126',
          'NB_A203',
          'NB_A205',
          'NB_A206',
          'NB_A207',
          'NB_A208',
          'NB_A301',
          'NB_A302',
          'NB_A303',
          'NB_A304',
          'NB_A401',
          'NB_A402',
          'NB_A403',
          'NB_A404',
'NB_A405',
'NB_A406',
'NB_A501',
'NB_A502',
'NB_A503',
'NB_A504',
'NB_A505',
'NB_A506',
'NB_A507',
'NB_D101',
'NB_D102',
'NB_D103',
'NB_D104',
'NB_D105',
'NB_D106',
'NB_D107',
'NB_D108',
'NB_D109',
'NB_D110',
'NB_D111',
'NB_D112',
'NB_D113',
'NB_D301',
'NB_D302',
'NB_D303',
'NB_D304',
'NB_D305',
'NB_D401',
'NB_D402',
'NB_D403',
'NB_D404',
'NB_D405',
'NB_D502',
'NB_D601',
'NB_D602',
'NB_D603',
'NB_D604',
'NB_D605',
'NB_D606',
'NB_D701',
'NB_D702',
'NB_D703',
'NB_D704',
'NB_D705',
'NB_D709',
'NB_E101',
'NB_E102',
'NB_E103',
 'NB_E106',
 'NB_F101',
 'NB_F101_NB_AIREJEU',
 'NB_F101_NB_COU',
 'NB_F101_NB_ECL',
 'NB_F102',
 'NB_F102_NB_AIREJEU',
 'NB_F102_NB_COU',
 'NB_F102_NB_ECL',
 'NB_F103',
 'NB_F103_NB_AIREJEU',
 'NB_F103_NB_COU',
 'NB_F103_NB_ECL',
 'NB_F104',
 'NB_F104_NB_AIREJEU',
 'NB_F104_NB_COU',
 'NB_F104_NB_ECL',
 'NB_F105',
 'NB_F105_NB_AIREJEU',
 'NB_F105_NB_COU',
 'NB_F105_NB_ECL',
 'NB_F106',
 'NB_F106_NB_AIREJEU',
 'NB_F106_NB_COU',
 'NB_F106_NB_ECL',
 'NB_F107',
 'NB_F107_NB_AIREJEU',
 'NB_F107_NB_COU',
 'NB_F107_NB_ECL',
 'NB_F108',
 'NB_F108_NB_AIREJEU',
 'NB_F108_NB_COU',
 'NB_F108_NB_ECL',
 'NB_F109',
 'NB_F109_NB_AIREJEU',
 'NB_F109_NB_COU',
 'NB_F109_NB_ECL',
 'NB_F110',
 'NB_F110_NB_AIREJEU',
 'NB_F110_NB_COU',
 'NB_F110_NB_ECL',
 'NB_F111',
 'NB_F111_NB_AIREJEU',
 'NB_F111_NB_COU',
 'NB_F111_NB_ECL',
 'NB_F112',
 'NB_F112_NB_AIREJEU',
 'NB_F112_NB_COU',
 'NB_F112_NB_ECL',
 'NB_F113',
 'NB_F113_NB_AIREJEU',
 'NB_F113_NB_COU',
 'NB_F113_NB_ECL',
 'NB_F114',
 'NB_F114_NB_AIREJEU',
 'NB_F114_NB_COU',
 'NB_F114_NB_ECL',
 'NB_F116',
 'NB_F116_NB_AIREJEU',
 'NB_F116_NB_COU',
 'NB_F116_NB_ECL',
 'NB_F117',
 'NB_F117_NB_AIREJEU',
 'NB_F117_NB_COU',
 'NB_F117_NB_ECL',
 'NB_F118',
 'NB_F118_NB_AIREJEU',
 'NB_F118_NB_COU',
 'NB_F118_NB_ECL',
 'NB_F119',
 'NB_F119_NB_AIREJEU',
 'NB_F119_NB_COU',
 'NB_F119_NB_ECL',
 'NB_F120',
 'NB_F120_NB_AIREJEU',
 'NB_F120_NB_COU',
 'NB_F120_NB_ECL',
 'NB_F121',
 'NB_F121_NB_AIREJEU',
 'NB_F121_NB_COU',
 'NB_F121_NB_ECL',
 'NB_F201',
 'NB_F201_NB_AIREJEU',
 'NB_F202',
 'NB_F202_NB_AIREJEU',
 'NB_F203',
 'NB_F203_NB_AIREJEU',
 'NB_F302',
 'NB_F302_NB_SALLES',
 'NB_F303',
 'NB_F303_NB_SALLES',
 'NB_F304',
 'NB_F305',
 'NB_G101',
 'NB_G102',
 'NB_G103',
 'NB_G104',
 'P10_EMPLT',
 'P10_POP',
 'P15_ACT1564',
 'P15_CHOM1564',
 'P15_EMPLT',
 'P15_EMPLT_SAL',
 'P15_LOG',
 'P15_LOGVAC',
 'P15_MEN',
 'P15_POP',
 'P15_POP1564',
 'P15_RP',
 'P15_RP_PROP',
 'P15_RSECOCC',
 'PIMP15',
 'Part.forêts.et.milieux.semi.naturels...2012...',
 'Part.protection.contractuelle...2017...',
 'Part.protection.forte...2017...',
 'Part.zones.humides.et.surfaces.en.eau...2012...',
 'REG',
 'SUPERF',
 'Superficie.forêts.et.milieux.semi.naturels...2012..ha.',
 'Superficie.protection.contractuelle...2017..ha.',
 'Superficie.protection.forte...2017..ha.',
 'Superficie.zones.humides.et.surfaces.en.eau...2012..ha.',
 'TP6015',
 'communes'}
```

```
In [30]: dict_var_groups["cat_min10_var"] = dict_var_groups["cat_min10_var"] | {'DEP', 'LIBGEO',
                                                                                  'communes'}
         dict_var_groups["cat_var"] = dict_var_groups["cat_var"] | {'DEP', 'LIBGEO', 'communes'}
```

```
In [31]: dict_var_groups["commune_var"] = commune_var
```

```
In [32]: df = MergeCommunesEnvi.loc[:,commune_var]
         commune_quant_var = set(df.select_dtypes(include=['float64']).columns)
         dict_var_groups["quant_var"] = dict_var_groups["quant_var"] | commune_quant_var
```

```
In [33]: cdv_2015_var = dict_var_groups['cdv_2015_var']
         cdv_2016_var = dict_var_groups['cdv_2016_var']
         cdv_2017_var = dict_var_groups['cdv_2017_var']
         cdv_2018_var = dict_var_groups['cdv_2018_var']
         cdv_2015_2018_var = dict_var_groups['cdv_2015_2018_var']
         cdv_2016_2018_var = dict_var_groups['cdv_2016_2018_var']
         cdv_2017_2018_var = dict_var_groups['cdv_2017_2018_var']

         scope_2015_var = cdv_2015_var | commune_var
         scope_2016_var = cdv_2016_var | commune_var
         scope_2017_var = cdv_2017_var | commune_var
         scope_2018_var = cdv_2018_var | commune_var
         scope_2015_2018_var = cdv_2015_2018_var | commune_var
         scope_2016_2018_var = cdv_2016_2018_var | commune_var
         scope_2017_2018_var = cdv_2017_2018_var | commune_var

         dict_var_groups["scope_2015_var"] = scope_2015_var
         dict_var_groups["scope_2016_var"] = scope_2016_var
         dict_var_groups["scope_2017_var"] = scope_2017_var
         dict_var_groups["scope_2018_var"] = scope_2018_var
         dict_var_groups["scope_2015_2018_var"] = scope_2015_2018_var
         dict_var_groups["scope_2016_2018_var"] = scope_2016_2018_var
         dict_var_groups["scope_2017_2018_var"] = scope_2017_2018_var
```

```
In [34]: dict_var_groups["indiv_semi_act_var"] = indiv_semi_act_var
         dict_var_groups["indiv_act_var"] = indiv_act_var
         dict_var_groups["admin_semi_act_var"] = admin_semi_act_var
         dict_var_groups["admin_act_var"] = admin_act_var
```

```
In [35]: com_var = dict_var_groups['com_var']
         tech_var = dict_var_groups['tech_var']
         text_var = dict_var_groups['text_var']
         bizz_var = dict_var_groups['bizz_var']

         cat_max9_var = dict_var_groups['cat_max9_var']
         quant_var = dict_var_groups['quant_var']

         exclusion = com_var | tech_var | bizz_var | text_var
         # suppresionn of quantitative variables with more than 200 NaN
         quant_null = np.sum(MergeCommunesEnvi.loc[:,quant_var].isnull())
         quant_var_kept = set(quant_null[quant_null < 200].index)

         usual_common_scope = ((cat_max9_var | quant_var_kept) & scope_2015_2018_var) - exclusio

         dict_var_groups["exclusion"] = exclusion
         dict_var_groups["usual_common_scope"] = usual_common_scope
```

1.1.5 Adding work on features of September

```
In [36]: # loadind xlsx file with agreement data
         file = path_data / Path("Base of Actionable Var. - Survey Data.xlsx")
         with Path.open(file, 'rb') as fp:
             agreement = pd.read_excel(fp,
                                     sheetname='List 1 Actionable Individual',
                                     parse_cols="C,H",
                                     index_col=0
                                    )
```

```
In [37]: agreement.head()
```

```
Out[37]:            Agreement
         Variables
         INTER6             4
         INTER              4
         ANNEEFUZ           4
         ANNEFUZ2           4
         COLLECTE           4
```

```
In [38]: cdv_actionable_individual_1 = set(agreement.loc[agreement.loc[:,"Agreement"]==1,:].inde
         cdv_actionable_individual_2 = set(agreement.loc[agreement.loc[:,"Agreement"]==2,:].inde
         cdv_actionable_individual_3 = set(agreement.loc[agreement.loc[:,"Agreement"]==3,:].inde
         cdv_actionable_individual_4 = set(agreement.loc[agreement.loc[:,"Agreement"]==4,:].inde
```

```
In [39]: dict_var_groups["cdv_actionable_individual_1"] = cdv_actionable_individual_1
         dict_var_groups["cdv_actionable_individual_2"] = cdv_actionable_individual_2
         dict_var_groups["cdv_actionable_individual_3"] = cdv_actionable_individual_3
         dict_var_groups["cdv_actionable_individual_4"] = cdv_actionable_individual_4
```

```
In [40]: # loadind xlsx file with agreement data
         file = path_data / Path("Base of Actionable Var. - Survey Data.xlsx")
         with Path.open(file, 'rb') as fp:
             agreement = pd.read_excel(fp,
                                       sheetname='List 2 Actionable Admin',
                                       parse_cols="C,H",
                                       index_col=0
                                      )
```

```
In [41]: agreement.head()
```

```
Out[41]:            Agreement
         Variables
         INTER6             4
         INTER              4
         ANNEEFUZ           4
         ANNEFUZ2           4
         COLLECTE           4
```

```
In [42]: cdv_actionable_admin_1 = set(agreement.loc[agreement.loc[:,"Agreement"]==1,:].index)
         cdv_actionable_admin_2 = set(agreement.loc[agreement.loc[:,"Agreement"]==2,:].index)
         cdv_actionable_admin_3 = set(agreement.loc[agreement.loc[:,"Agreement"]==3,:].index)
         cdv_actionable_admin_4 = set(agreement.loc[agreement.loc[:,"Agreement"]==4,:].index)
         cdv_actionable_admin_5 = set(agreement.loc[agreement.loc[:,"Agreement"]==5,:].index)
```

```
In [43]: dict_var_groups["cdv_actionable_admin_1"] = cdv_actionable_admin_1
         dict_var_groups["cdv_actionable_admin_2"] = cdv_actionable_admin_2
         dict_var_groups["cdv_actionable_admin_3"] = cdv_actionable_admin_3
         dict_var_groups["cdv_actionable_admin_4"] = cdv_actionable_admin_4
         dict_var_groups["cdv_actionable_admin_5"] = cdv_actionable_admin_5
```

```
In [44]: # loadind xlsx file with agreement data
         file = path_data / Path("Base Admin Action. Var. - Recreation.xlsx")
         with Path.open(file, 'rb') as fp:
             agreement = pd.read_excel(fp,
                                       sheetname='Actionable Variables',
                                       parse_cols="B,E",
                                       index_col=0,
                                       skiprows=[0,1]
                                      )
```

```
In [45]: agreement.head()
```

```
Out[45]:                         Agreement
         Variable
         CODGEO                          4
         Libellé commune ou ARM          4
         Région                          4
         Département                     4
         Bassin de natation              1
```

```
In [46]: insee_recreation_actionable_admin_1 = set(agreement.loc[agreement.loc[:,"Agreement"]==1
         insee_recreation_actionable_admin_2 = set(agreement.loc[agreement.loc[:,"Agreement"]==2
         insee_recreation_actionable_admin_3 = set(agreement.loc[agreement.loc[:,"Agreement"]==3
         insee_recreation_actionable_admin_4 = set(agreement.loc[agreement.loc[:,"Agreement"]==4
         insee_recreation_actionable_admin_5 = set(agreement.loc[agreement.loc[:,"Agreement"]==5
```

```
In [47]: dict_var_groups["insee_recreation_actionable_admin_1"] = insee_recreation_actionable_ad
         dict_var_groups["insee_recreation_actionable_admin_2"] = insee_recreation_actionable_ad
         dict_var_groups["insee_recreation_actionable_admin_3"] = insee_recreation_actionable_ad
         dict_var_groups["insee_recreation_actionable_admin_4"] = insee_recreation_actionable_ad
         dict_var_groups["insee_recreation_actionable_admin_5"] = insee_recreation_actionable_ad
```

```
In [48]: # loadind xlsx file with agreement data
         file = path_data / Path("Base Admin Action. Var. - Environment.xlsx")
         with Path.open(file, 'rb') as fp:
             agreement = pd.read_excel(fp,
                                       sheetname='Actionable Variables',
                                       parse_cols="B,E",
                                       index_col=0,
                                       skiprows=[0,1,2,3,4]
                                      )
```

```
In [49]: agreement.head()
```

```
Out[49]:                                                 Agreement
         Variable
         Code                                                    4
         communes                                                4
         Superficie protection forte - 2017 (ha)                 4
         Part protection forte - 2017 (%)                        1
         Superficie protection contractuelle - 2017 (ha)         4
```

```
In [50]: insee_environment_actionable_admin_1 = set(agreement.loc[agreement.loc[:,"Agreement"]==
         insee_environment_actionable_admin_2 = set(agreement.loc[agreement.loc[:,"Agreement"]==
         insee_environment_actionable_admin_3 = set(agreement.loc[agreement.loc[:,"Agreement"]==
         insee_environment_actionable_admin_4 = set(agreement.loc[agreement.loc[:,"Agreement"]==
         insee_environment_actionable_admin_5 = set(agreement.loc[agreement.loc[:,"Agreement"]==
```

```
In [51]: dict_var_groups["insee_environment_actionable_admin_1"] = insee_environment_actionable_
         dict_var_groups["insee_environment_actionable_admin_2"] = insee_environment_actionable_
         dict_var_groups["insee_environment_actionable_admin_3"] = insee_environment_actionable_
         dict_var_groups["insee_environment_actionable_admin_4"] = insee_environment_actionable_
         dict_var_groups["insee_environment_actionable_admin_5"] = insee_environment_actionable_
```

```
In [52]: # loadind xlsx file with agreement data
         file = path_data / Path("Base Admin Action. Var. - Demographics.xlsx")
         with Path.open(file, 'rb') as fp:
             agreement = pd.read_excel(fp,
                                       sheetname='Actionable Variables',
                                       parse_cols="D,G",
                                       index_col=0,
                                       skiprows=[0,1,2,3,4,5,6]
                                      )
```

```
In [53]: agreement.head()
```

```
Out[53]:                                                     Agreement
         VAR_LIB_LONG
         Code du département suivi du numéro de commune ...          4
         Libellé de la commune ou de l'arrondissement mu...          4
         Région                                                      4
         Département                                                 4
         Population en 2015                                          3
```

```
In [54]: insee_demographics_actionable_admin_1 = set(agreement.loc[agreement.loc[:,"Agreement"]=
         insee_demographics_actionable_admin_2 = set(agreement.loc[agreement.loc[:,"Agreement"]=
         insee_demographics_actionable_admin_3 = set(agreement.loc[agreement.loc[:,"Agreement"]=
         insee_demographics_actionable_admin_4 = set(agreement.loc[agreement.loc[:,"Agreement"]=
         insee_demographics_actionable_admin_5 = set(agreement.loc[agreement.loc[:,"Agreement"]=
```

```
In [55]: dict_var_groups["insee_demographics_actionable_admin_1"] = insee_demographics_actionabl
         dict_var_groups["insee_demographics_actionable_admin_2"] = insee_demographics_actionabl
         dict_var_groups["insee_demographics_actionable_admin_3"] = insee_demographics_actionabl
         dict_var_groups["insee_demographics_actionable_admin_4"] = insee_demographics_actionabl
         dict_var_groups["insee_demographics_actionable_admin_5"] = insee_demographics_actionabl
```

```
In [56]: [k for k in dict_var_groups.keys()]
```

```
Out[56]: ['cdv_2015_var',
          'cdv_2016_var',
          'cdv_2017_var',
          'cdv_2018_var',
          'cdv_2015_2018_var',
          'cdv_2016_2018_var',
          'cdv_2017_2018_var',
          'pred_var',
          'tech_var',
          'com_var',
          'text_var',
          'bizz_var',
          'cat_var',
          'cat_max9_var',
          'cat_min10_var',
          'quant_var',
          'commune_var',
          'scope_2015_var',
          'scope_2016_var',
          'scope_2017_var',
          'scope_2018_var',
          'scope_2015_2018_var',
          'scope_2016_2018_var',
          'scope_2017_2018_var',
          'indiv_semi_act_var',
          'indiv_act_var',
          'admin_semi_act_var',
          'admin_act_var',
          'exclusion',
          'usual_common_scope',
          'cdv_actionable_individual_1',
          'cdv_actionable_individual_2',
          'cdv_actionable_individual_3',
          'cdv_actionable_individual_4',
          'cdv_actionable_admin_1',
          'cdv_actionable_admin_2',
          'cdv_actionable_admin_3',
          'cdv_actionable_admin_4',
          'cdv_actionable_admin_5',
          'insee_recreation_actionable_admin_1',
          'insee_recreation_actionable_admin_2',
          'insee_recreation_actionable_admin_3',
          'insee_recreation_actionable_admin_4',
          'insee_recreation_actionable_admin_5',
          'insee_environment_actionable_admin_1',
          'insee_environment_actionable_admin_2',
          'insee_environment_actionable_admin_3',
          'insee_environment_actionable_admin_4',
          'insee_environment_actionable_admin_5',
          'insee_demographics_actionable_admin_1',
          'insee_demographics_actionable_admin_2',
          'insee_demographics_actionable_admin_3',
          'insee_demographics_actionable_admin_4',
          'insee_demographics_actionable_admin_5']
```

```
In [57]: filename = path_dump / Path("dict_var_groups.sav")
         with open(filename, 'wb') as fp:
              pickle.dump(dict_var_groups,fp,pickle.HIGHEST_PROTOCOL)
```

Dataset

```
In [58]: df = MergeCommunesEnvi.loc[:,:]
         df.loc[:,cdv_ssfmt.columns] = cdv_ssfmt.loc[:,:]
         df = df.loc[:,usual_common_scope]
         df.loc[:,(cat_var & usual_common_scope) - {"HEUREUX"}] = cdv.loc[:,(cat_var & usual_com


         dataset = pd.get_dummies(
             df,
             columns=(cat_var & usual_common_scope) - {"HEUREUX"},
             dummy_na = True,
             drop_first=1
         )

         print(f"{dataset.shape[1]} columns after encoding of {len((cat_var & usual_common_scope
         categorial variables in {len((cat_var & usual_common_scope))-1+dataset.shape[1]-df.shap
          binary variables (K-1 one hot encoding)")
```

```
810 columns after encoding of 150categorial variables in 553 binary variables (K-1 one hot encod
```

```
In [59]: # saving dataset data
         file = path_data / Path("dataset.csv")
         with Path.open(file, 'w') as fp:
             dataset.to_csv(fp,  encoding='utf-8')
```

```
In [60]: idx_2017_2018 = MergeCommunesEnvi.loc[MergeCommunesEnvi['ANNEEFUZ'].isin([39,40]),:].in
         df = MergeCommunesEnvi.loc[idx_2017_2018,:]
         df.loc[:,cdv_ssfmt.columns] = cdv_ssfmt.loc[idx_2017_2018,:]


         f_2017_2018 = ((cat_max9_var | quant_var_kept) & scope_2017_2018_var) - exclusion

         df = df.loc[:,f_2017_2018]
         df.loc[:,(cat_var & f_2017_2018) - {"HEUREUX"}] = cdv.loc[idx_2017_2018,(cat_var & f_20


         dataset_2017_2018 = pd.get_dummies(
             df,
             columns=(cat_var & usual_common_scope) - {"HEUREUX"},
             dummy_na = True,
             drop_first=1
         )

         print(f"{dataset_2017_2018.shape[1]} columns after encoding of {len((cat_var & f_2017_2
         categorial variables in {len((cat_var & f_2017_2018))-1+dataset_2017_2018.shape[1]-df.s
         binary variables (K-1 one hot encoding)")
```

```
796 columns after encoding of 159 categorial variables in 539 binary variables (K-1 one hot enco
```

```
In [61]: # saving dataset data
         file = path_data / Path("dataset_2017_2018.csv")
         with Path.open(file, 'w') as fp:
             dataset_2017_2018.to_csv(fp,  encoding='utf-8')
```

Construction of usefull feature sets Including ... Lasso or other feature selection methods, ...

```
In [62]: dict_features_sets = dict()
```

```
In [63]: usual_common_features  = set(dataset.columns)
         dict_features_sets['usual_common_features'] = usual_common_features
```

```
In [64]: df = MergeCommunesEnvi.loc[:,:]
         df.loc[:,cdv_ssfmt.columns] = cdv_ssfmt.loc[:,:]
         scope = indiv_act_var & usual_common_scope
         df = df.loc[:,scope]
         df.loc[:,(cat_var & scope) - {"HEUREUX"}] = cdv.loc[:,(cat_var & scope) - {"HEUREUX"}]

         df_dummies = pd.get_dummies(
             df,
             columns=(cat_var & scope),
             dummy_na = True,
             drop_first=1
         )

         print(f"{df_dummies.shape[1]} columns after encoding of {len((cat_var & scope))} catego
         variables in {len((cat_var & scope))+df_dummies.shape[1]-df.shape[1]} binary variables
         (K-1 one hot encoding)")

         indiv_act_features = set(df_dummies.columns)
         dict_features_sets['indiv_act_features'] = indiv_act_features
```

```
50 columns after encoding of 13 categorial variables in 40 binary variables (K-1 one hot encodin
```

```
In [65]: df = MergeCommunesEnvi.loc[:,:]
         df.loc[:,cdv_ssfmt.columns] = cdv_ssfmt.loc[:,:]
         scope = indiv_semi_act_var & usual_common_scope
```

```
        df = df.loc[:,scope]
        df.loc[:,(cat_var & scope) - {"HEUREUX"}] = cdv.loc[:,(cat_var & scope) - {"HEUREUX"}]

        df_dummies = pd.get_dummies(
            df,
            columns=(cat_var & scope),
            dummy_na = True,
            drop_first=1
        )

        print(f"{df_dummies.shape[1]} columns after encoding of {len((cat_var & scope))} catego
        variables in {len((cat_var & scope))+df_dummies.shape[1]-df.shape[1]} binary variables
        (K-1 one hot encoding)")

        indiv_semi_act_features = set(df_dummies.columns)
        dict_features_sets['indiv_semi_act_features'] = indiv_semi_act_features
```

```
73 columns after encoding of 17 categorial variables in 60 binary variables (K-1 one hot encodin
```

```
In [66]: df = MergeCommunesEnvi.loc[:,:]
        df.loc[:,cdv_ssfmt.columns] = cdv_ssfmt.loc[:,:]
        scope = admin_act_var & usual_common_scope
        df = df.loc[:,scope]
        df.loc[:,(cat_var & scope) - {"HEUREUX"}] = cdv.loc[:,(cat_var & scope) - {"HEUREUX"}]

        df_dummies = pd.get_dummies(
            df,
            columns=(cat_var & scope),
            dummy_na = True,
            drop_first=1
        )

        print(f"{df_dummies.shape[1]} columns after encoding of {len((cat_var & scope))} catego
        variables in {len((cat_var & scope))+df_dummies.shape[1]-df.shape[1]} binary variables
        (K-1 one hot encoding)")

        admin_act_features = set(df_dummies.columns)
        dict_features_sets['admin_act_features'] = admin_act_features
```

```
13 columns after encoding of 3 categorial variables in 9 binary variables (K-1 one hot encoding)
```

```
In [67]: df = MergeCommunesEnvi.loc[:,:]
        df.loc[:,cdv_ssfmt.columns] = cdv_ssfmt.loc[:,:]
        scope = admin_semi_act_var & usual_common_scope
        df = df.loc[:,scope]
        df.loc[:,(cat_var & scope) - {"HEUREUX"}] = cdv.loc[:,(cat_var & scope) - {"HEUREUX"}]
```

```
df_dummies = pd.get_dummies(
    df,
    columns=(cat_var & scope),
    dummy_na = True,
    drop_first=1
)

print(f"{df_dummies.shape[1]} columns after encoding of {len((cat_var & scope))} catego
variables in {len((cat_var & scope))+df_dummies.shape[1]-df.shape[1]} binary variables
(K-1 one hot encoding)")

admin_semi_act_features = set(df_dummies.columns)
dict_features_sets['admin_semi_act_features'] = admin_semi_act_features
```

```
25 columns after encoding of 6 categorial variables in 20 binary variables (K-1 one hot encoding
```

```
In [68]: # Adding work of variable workshop
         df = MergeCommunesEnvi.loc[:,:]
         df.loc[:,cdv_ssfmt.columns] = cdv_ssfmt.loc[:,:]
         scope = (cdv_actionable_individual_1) & usual_common_scope
         df = df.loc[:,scope]
         df.loc[:,(cat_var & scope) - {"HEUREUX"}] = cdv.loc[:,(cat_var & scope) - {"HEUREUX"}]

         df_dummies = pd.get_dummies(
             df,
             columns=(cat_var & scope),
             dummy_na = True,
             drop_first=1
         )

         print(f"{df_dummies.shape[1]} columns after encoding of {len((cat_var & scope))} catego
         variables in {len((cat_var & scope))+df_dummies.shape[1]-df.shape[1]} binary variables
         (K-1 one hot encoding)")

         cdv_actionable_individual_1_features = set(df_dummies.columns)
         dict_features_sets['cdv_actionable_individual_1_features'] = cdv_actionable_individual_
```

```
193 columns after encoding of 57 categorial variables in 163 binary variables (K-1 one hot encod
```

```
In [69]: # Adding work of variable workshop
         df = MergeCommunesEnvi.loc[:,:]
         df.loc[:,cdv_ssfmt.columns] = cdv_ssfmt.loc[:,:]
         scope = (cdv_actionable_individual_2) & usual_common_scope
         df = df.loc[:,scope]
         df.loc[:,(cat_var & scope) - {"HEUREUX"}] = cdv.loc[:,(cat_var & scope) - {"HEUREUX"}]

         df_dummies = pd.get_dummies(
             df,
             columns=(cat_var & scope),
             dummy_na = True,
             drop_first=1
         )

         print(f"{df_dummies.shape[1]} columns after encoding of {len((cat_var & scope))} catego
         variables in {len((cat_var & scope))+df_dummies.shape[1]-df.shape[1]} binary variables
         (K-1 one hot encoding)")

         cdv_actionable_individual_2_features = set(df_dummies.columns)
         dict_features_sets['cdv_actionable_individual_2_features'] = cdv_actionable_individual_
```

```
230 columns after encoding of 54 categorial variables in 210 binary variables (K-1 one hot encod
```

```
In [70]: # Adding work of variable workshop
         df = MergeCommunesEnvi.loc[:,:]
         df.loc[:,cdv_ssfmt.columns] = cdv_ssfmt.loc[:,:]
         scope = (cdv_actionable_admin_1) & usual_common_scope
         df = df.loc[:,scope]
         df.loc[:,(cat_var & scope) - {"HEUREUX"}] = cdv.loc[:,(cat_var & scope) - {"HEUREUX"}]

         df_dummies = pd.get_dummies(
             df,
             columns=(cat_var & scope),
             dummy_na = True,
             drop_first=1
         )

         print(f"{df_dummies.shape[1]} columns after encoding of {len((cat_var & scope))} catego
         variables in {len((cat_var & scope))+df_dummies.shape[1]-df.shape[1]} binary variables
         (K-1 one hot encoding)")

         cdv_actionable_admin_1_features = set(df_dummies.columns)
         dict_features_sets['cdv_actionable_admin_1_features'] = cdv_actionable_admin_1_features
```

```
201 columns after encoding of 59 categorial variables in 168 binary variables (K-1 one hot encod
```

```
In [71]: # Adding work of variable workshop
         df = MergeCommunesEnvi.loc[:,:]
         df.loc[:,cdv_ssfmt.columns] = cdv_ssfmt.loc[:,:]
         scope = (cdv_actionable_admin_2) & usual_common_scope
         df = df.loc[:,scope]
         df.loc[:,(cat_var & scope) - {"HEUREUX"}] = cdv.loc[:,(cat_var & scope) - {"HEUREUX"}]

         df_dummies = pd.get_dummies(
             df,
             columns=(cat_var & scope),
             dummy_na = True,
             drop_first=1
         )

         print(f"{df_dummies.shape[1]} columns after encoding of {len((cat_var & scope))} catego
         variables in {len((cat_var & scope))+df_dummies.shape[1]-df.shape[1]} binary variables
         (K-1 one hot encoding)")

         cdv_actionable_admin_2_features = set(df_dummies.columns)
         dict_features_sets['cdv_actionable_admin_2_features'] = cdv_actionable_admin_2_features
```

```
202 columns after encoding of 47 categorial variables in 188 binary variables (K-1 one hot encod
```

```
In [72]: # Adding work of variable workshop
         df = MergeCommunesEnvi.loc[:,:]
         df.loc[:,cdv_ssfmt.columns] = cdv_ssfmt.loc[:,:]
         scope = (cdv_actionable_admin_3) & usual_common_scope
         df = df.loc[:,scope]
         df.loc[:,(cat_var & scope) - {"HEUREUX"}] = cdv.loc[:,(cat_var & scope) - {"HEUREUX"}]

         df_dummies = pd.get_dummies(
             df,
             columns=(cat_var & scope),
             dummy_na = True,
             drop_first=1
         )

         print(f"{df_dummies.shape[1]} columns after encoding of {len((cat_var & scope))} catego
         variables in {len((cat_var & scope))+df_dummies.shape[1]-df.shape[1]} binary variables
         (K-1 one hot encoding)")

         cdv_actionable_admin_3_features = set(df_dummies.columns)
         dict_features_sets['cdv_actionable_admin_3_features'] = cdv_actionable_admin_3_features
```

```
60 columns after encoding of 15 categorial variables in 57 binary variables (K-1 one hot encodin
```

```
In [73]: # Adding work of variable workshop
         # insee_demographics_actionable_admin_1 insee_demographics_actionable_admin_2 ...
         # variable name wil have to be restated
```

```
In [74]: filename = path_dump / Path("dict_features_sets.sav")
         with open(filename, 'wb') as fp:
              pickle.dump(dict_features_sets,fp,pickle.HIGHEST_PROTOCOL)
```

Feature selection and and results recording

```
In [75]: # reducing problem to a 2 class classification problem
         df = dataset.loc[:,:]
         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

         # treating remaining missing values
         features = set(df.columns.drop(['HEUREUX', 'HEUREUX_CLF']))
         df = df.loc[:,features | {"HEUREUX_CLF"}].dropna()
```

```
In [76]: X = df.loc[:,features]
         y = df["HEUREUX_CLF"]

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


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

         print(f"Number exemple: {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: 10445
- training set: 8356
- test set: 2089
Number of features: p=809
Number of class: 2
class 0 : 35.1%
class 1 : 64.9%
```

```
In [77]: clf = LinearSVC(C=0.01,
                         class_weight='balanced',
                         dual=False,
                         random_state=42 )
         step = 0.05
```

```
In [78]: for n_features_to_select in [100,50,20,10]:
             startTime = time.time()
             print(f"number of features to select : {n_features_to_select}")
             selector = RFE(estimator=clf, n_features_to_select=n_features_to_select, step=step)
```

```
            selector.fit(X_train, y_train)
            print(f"Optimal support of size {n_features_to_select} found in {time.time() - star
            key = "RFE_LinearSVC_" + str(n_features_to_select) + "_features"
            dict_features_sets[key] = set(X.loc[:,selector.support_].columns)
```

```
number of features to select : 100
Optimal support of size 100 found in 139.0 s
number of features to select : 50
Optimal support of size 50 found in 131.6 s
number of features to select : 20
Optimal support of size 20 found in 132.8 s
number of features to select : 10
Optimal support of size 10 found in 130.1 s
```

```
In [79]: params = {'max_features' :'sqrt', 'random_state' : 32,
                   'min_samples_split' : 2, 'class_weight' : 'balanced',
                   'n_estimators' : 128,
                   'max_depth' : 8}
         clf = RandomForestClassifier(**params)
         step = 0.05
```

```
In [80]: for n_features_to_select in [100,50,20,10]:
             startTime = time.time()
             print(f"number of features to select : {n_features_to_select}")
             selector = RFE(estimator=clf, n_features_to_select=n_features_to_select, step=step)
             selector.fit(X_train, y_train)
             print(f"Optimal support of size {n_features_to_select} found in {time.time() - star
             key = "RFE_RandomForestClassifier_" + str(n_features_to_select) + "_features"
             dict_features_sets[key] = set(X.loc[:,selector.support_].columns)
```

```
number of features to select : 100
Optimal support of size 100 found in 67.7 s
number of features to select : 50
Optimal support of size 50 found in 70.0 s
number of features to select : 20
Optimal support of size 20 found in 70.3 s
number of features to select : 10
Optimal support of size 10 found in 69.7 s
```

```
In [81]: clf = LogisticRegression(C=0.01,
                                  penalty='l1',
                                  class_weight='balanced',
                                  random_state=42)
         step = 0.05
```

```
In [82]: for n_features_to_select in [100,50,20,10]:
             startTime = time.time()
             print(f"number of features to select : {n_features_to_select}")
             selector = RFE(estimator=clf, n_features_to_select=n_features_to_select, step=step)
             selector.fit(X_train, y_train)
             print(f"Optimal support of size {n_features_to_select} found in {time.time() - star
             key = "RFE_LogisticRegression_" + str(n_features_to_select) + "_features"
             dict_features_sets[key] = set(X.loc[:,selector.support_].columns)
```

```
number of features to select : 100
Optimal support of size 100 found in 7.4 s
number of features to select : 50
Optimal support of size 50 found in 7.1 s
number of features to select : 20
Optimal support of size 20 found in 7.1 s
number of features to select : 10
Optimal support of size 10 found in 7.1 s
```

SelectFromModel

```
In [83]: clf = LinearSVC(C=0.01, penalty="l1", dual=False, class_weight='balanced' ).fit(X_train
         model = SelectFromModel(clf, prefit=True)
         dict_features_sets['SelectFromModel_LinearSCV_features'] = set(X.loc[:,model.get_suppor
```

```
In [84]: clf = LogisticRegression(C=0.01, penalty="l1",class_weight='balanced',random_state=42 )
         model = SelectFromModel(clf, prefit=True)
         dict_features_sets['SelectFromModel_LogisticRegression_features'] = set(X.loc[:,model.g
```

```
In [85]: filename = path_dump / Path("dict_features_sets.sav")
         with open(filename, 'wb') as fp:
              pickle.dump(dict_features_sets,fp,pickle.HIGHEST_PROTOCOL)
```

```
In [87]: [k for k in dict_features_sets.keys()]
```

```
Out[87]: ['usual_common_features',
          'indiv_act_features',
          'indiv_semi_act_features',
          'admin_act_features',
          'admin_semi_act_features',
          'cdv_actionable_individual_1_features',
          'cdv_actionable_individual_2_features',
          'cdv_actionable_admin_1_features',
          'cdv_actionable_admin_2_features',
          'cdv_actionable_admin_3_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']
```