Deduplication using machine learning

February 6, 2018

1 Example of Deduplication

```
In [30]: import deduplication as dep
         reload(dep);
```

1.1 1. Loading Data

1.1.1 Loading sample data

```
In [31]: df_input_records = pd.read_csv('df_impairs.csv',index_col=0,dtype={'duns':str,'postal
         df_target_records = pd.read_csv('df_pairs.csv',index_col=0,dtype={'duns':str,'postalc
         df_input_records.sample(3)
```

```
Out[31]:                                                          name        duns  \
         gid
         661a0913-d82f-4186-b191-58a93c06ba0c  phoenix contact hmi-ipc         NaN
         774bcf85-7a03-4a6d-9ec3-f5474b5e996d              aviasport sa  464418029
         efb815f9-ea51-4728-babb-fdd9c294981d        pro-idee gmbh co kg  312865546

                                                       city postalcode  \
         gid
         661a0913-d82f-4186-b191-58a93c06ba0c  filderstadt      70794
         774bcf85-7a03-4a6d-9ec3-f5474b5e996d  tres cantos      28760
         efb815f9-ea51-4728-babb-fdd9c294981d       aachen      52070

                                                         street country_code
         gid
         661a0913-d82f-4186-b191-58a93c06ba0c       29 kurze str           DE
         774bcf85-7a03-4a6d-9ec3-f5474b5e996d  11 calle almazara           ES
         efb815f9-ea51-4728-babb-fdd9c294981d       gut-dmme-str           DE
```

1.1.2 Cleaning that data

```
In [32]: from preprocessing import clean_db
         for x in [df_input_records,df_target_records]:
             x = clean_db(x)
         df_target_records.sample(3)
```

Out[32]:

```
                                                         name       duns  \
gid
b537e8e0-1a92-4694-8ef7-c43c4b7a208f    jamara modelltechnik  322899675
e8695ac0-8ef8-4bd1-bbfb-36009497d10a      cadilac laser gmbh       None
941ec1ab-0b44-4a66-abd6-69eead48b1ca  airtanker services ltd       None

                                             city postalcode  \
gid
b537e8e0-1a92-4694-8ef7-c43c4b7a208f  aichstetten      88317
e8695ac0-8ef8-4bd1-bbfb-36009497d10a     albstadt      72459
941ec1ab-0b44-4a66-abd6-69eead48b1ca    carterton       ox18

                                                         street  \
gid
b537e8e0-1a92-4694-8ef7-c43c4b7a208f                5 am lauerbhl
e8695ac0-8ef8-4bd1-bbfb-36009497d10a                    herderstr
941ec1ab-0b44-4a66-abd6-69eead48b1ca  airtanker hub raf brize norton

                                     country_code      name_wostopwords  \
gid
b537e8e0-1a92-4694-8ef7-c43c4b7a208f           DE  jamara modelltechnik
e8695ac0-8ef8-4bd1-bbfb-36009497d10a           DE         cadilac laser
941ec1ab-0b44-4a66-abd6-69eead48b1ca           GB             airtanker

                                                 street_wostopwords  \
gid
b537e8e0-1a92-4694-8ef7-c43c4b7a208f                  lauerbhl 5 am
e8695ac0-8ef8-4bd1-bbfb-36009497d10a                         herder
941ec1ab-0b44-4a66-abd6-69eead48b1ca  raf airtanker hub norton brize

                                     name_acronym postalcode_1stdigit  \
gid
b537e8e0-1a92-4694-8ef7-c43c4b7a208f           jm                   8
e8695ac0-8ef8-4bd1-bbfb-36009497d10a          clg                   7
941ec1ab-0b44-4a66-abd6-69eead48b1ca          asl                   o

                                     postalcode_2digits  name_len  \
gid
b537e8e0-1a92-4694-8ef7-c43c4b7a208f                 88        20
e8695ac0-8ef8-4bd1-bbfb-36009497d10a                 72        18
941ec1ab-0b44-4a66-abd6-69eead48b1ca                 ox        22

                                      hasairbusname  isbigcity
gid
b537e8e0-1a92-4694-8ef7-c43c4b7a208f              0          0
e8695ac0-8ef8-4bd1-bbfb-36009497d10a              0          0
941ec1ab-0b44-4a66-abd6-69eead48b1ca              0          0
```

2 Machine Learning - based deduplication

2.1 Creating a training table for the decision model

2.1.1 Creating a side-by-side comparison table for manual labelling

using results from a rule-based decision model, for example (see below)

2.1.2 Loading a supervised learning table

```
In [33]: supervised_learning=pd.read_excel('supervised_table.xlsx')
         nix=1000
         s_inputs=supervised_learning['ix_source'].iloc[:nix]
         s_targets=supervised_learning['ix_target'].iloc[:nix]
         s_true=supervised_learning['y_true'].iloc[:nix]
```

2.1.3 Creating the training table

```
In [34]: dummymodel=dep.TrainerModel(scoredict={'fuzzy':['name','name_wostopwords',
                                                          'street','street_wostopwords',
                                                          'city'],
                                                 'exact':['duns','country_code'],
                                                 'token':['name','name_wostopwords',
                                                          'street','street_wostopwords'],
                                                 'acronym':['name','name_wostopwords']})
         sur=dep.Suricate(input_records=df_input_records,
                          target_records=df_target_records,
                          model=dummymodel)
         training_table=sur.build_training_table(inputs=s_inputs,targets=s_targets,y_true=s_tru
         #x2=sur.chain_build_labelled_table(inputs=s_inputs,targets=s_targets)
         print(training_table['y_true'].value_counts())
         training_table.sample(3)

0    916
1     84
Name: y_true, dtype: int64


Out[34]:      city_fuzzyscore  country_code_exactscore  duns_exactscore  \
         565             0.12                        1                0
         209             0.31                        1               -1
         188             0.00                        1               -1

              name_acronymscore  name_fuzzyscore  name_tokenscore  \
         565                0.0             0.41         0.333333
         209                0.0             0.49         0.333333
         188                0.0             0.50         0.333333

              name_wostopwords_acronymscore  name_wostopwords_fuzzyscore  \
```

```
565                           -1.0                        0.24
209                           -1.0                        0.32
188                           -1.0                        0.11

     name_wostopwords_tokenscore  street_fuzzyscore  street_tokenscore  \
565                          0.0               0.45                0.0
209                          0.0               0.52                0.0
188                          0.0               0.44                0.0

     street_wostopwords_fuzzyscore  street_wostopwords_tokenscore  y_true
565                           0.39                            0.0       0
209                           0.38                            0.0       0
188                           0.32                            0.0       0
```

2.1.4 Train the decision model

```
In [35]: X_train = training_table.iloc[:,:-1].astype(float)
         y_train= training_table.iloc[:,-1].astype(int)
```

```
In [36]: evaluator=dep.MLEvaluationModel()
         evaluator.fit(X=X_train,y=y_train)
```

```
shape of training table  (1000, 13)
number of positives in table 84
precision score on training data: 1.0
recall score on training data: 1.0
time elapsed 3.607531 seconds
```

2.1.5 Adding filtering rules to speed up the process (Optional)

filter on records that match exactly the country code, or that match the duns number

```
In [37]: filterdict={'all':['country_code'],
                     'any':['duns']}
```

from those filtered records, filter on records who have a roughly similar name or address, or share the same duns

```
In [38]: intermediate_thresholds={'name_wostopwords_fuzzyscore':0.6,'street_wostopwords_fuzzys
```

2.2 Launching the deduplication

```
In [39]: sur=dep.Suricate(input_records=df_input_records,target_records=df_target_records,
                          filterdict=filterdict,
                          intermediate_thresholds=intermediate_thresholds,
                          model=evaluator)
```

2.2.1 Possibility 1: return only good matches (for run mode)

```
In [40]: res=sur.start_linkage()
         df=sur.format_results(res,display=['name','street','duns','country_code'],fuzzyscorec
         df.sample(5)
```

```
starting deduplication at 2018-02-06 09:41:13.010887
1 of 10 inputs records deduplicated | found 0 of 1 max possible matches | time elapsed 0.25316
2 of 10 inputs records deduplicated | found 0 of 1 max possible matches | time elapsed 0.23138
3 of 10 inputs records deduplicated | found 0 of 1 max possible matches | time elapsed 0.25035
4 of 10 inputs records deduplicated | found 1 of 1 max possible matches | time elapsed 0.24468
5 of 10 inputs records deduplicated | found 1 of 1 max possible matches | time elapsed 0.22009
6 of 10 inputs records deduplicated | found 1 of 1 max possible matches | time elapsed 0.24675
7 of 10 inputs records deduplicated | found 1 of 1 max possible matches | time elapsed 0.25994
8 of 10 inputs records deduplicated | found 0 of 1 max possible matches | time elapsed 0.23812
9 of 10 inputs records deduplicated | found 1 of 1 max possible matches | time elapsed 0.19817
10 of 10 inputs records deduplicated | found 1 of 1 max possible matches | time elapsed 0.26408
finished work at 2018-02-06 09:41:15.419575
```

```
Out[40]:                                ix_source                             ix_target  \
         1  ff973ba5-ab42-42e0-8244-6aa82de46691  c3200b89-b646-4b61-affb-76023e5915ef
         3  e2a2da69-3aa4-44e2-ae5b-4bbd2cbdc238  c906f2e3-bd4c-4785-9d60-95f19579a04c
         2  6097f4c6-8515-41fb-b5e5-549c81140848  f704e1f0-b240-4461-a741-b41b5c30b476
         4  21c09dde-cff3-4c45-a2f1-46a99e6e1587  1b932c6a-3719-4f78-ba6e-c4ffdf0cc344
         0  5ff704ee-399e-4fbd-b604-51b6ced944dd  af8133f8-361f-494e-92dc-ab3c72637d56

                                 name_source                          name_target  \
         1                               acm                                  acm
         3           botschaft afghanistan                botschaft afghanistan
         2  bildungswerk der wirtschaft hamburg  bildungswerk der wirtschaft hamburg
         4       hatfield and dawson consulting       hatfield and dawson consulting
         0                     berlinzeppelin                       berlinzeppelin

           country_code_source country_code_target     street_source     street_target  \
         1                  FR                  FR  9 rue de la gare  9 rue de la gare
         3                  DE                  DE   3 taunusstraaye        3 taunusstr
         2                  DE                  DE   10 kapstadtring   10 kapstadtring
         4                  US                  US   greenwood ave n   greenwood ave n
         0                  DE                  DE  4 rottweiler str  4 rottweiler str

           duns_source duns_target  name_fuzzyscore  street_fuzzyscore  avg_fuzzyscore  \
         1   380071407        None              1.0               1.00           1.000
         3        None        None              1.0               0.85           0.925
         2        None        None              1.0               1.00           1.000
         4   099615556        None              1.0               1.00           1.000
         0        None        None              1.0               1.00           1.000

           duns_exactscore  n_exactmatches
```

```
1            None                 0
3            None                 0
2            None                 0
4            None                 0
0            None                 0
```

2.2.2 Possibility 2: return a probability vector to build a supervised learning table

```
In [41]: # return the 5 most probable matches of the query and the associated probabilities
         res=sur.start_linkage(n_matches_max=5,with_proba=True)
         df=sur.format_results(res,with_proba=True,display=['name','street','duns','country_co
         df.sample(3)
```

```
starting deduplication at 2018-02-06 09:41:15.472593
1 of 10 inputs records deduplicated | found 1 of 5 max possible matches | time elapsed 0.30197
2 of 10 inputs records deduplicated | found 1 of 5 max possible matches | time elapsed 0.50133
3 of 10 inputs records deduplicated | found 2 of 5 max possible matches | time elapsed 0.72946
4 of 10 inputs records deduplicated | found 1 of 5 max possible matches | time elapsed 0.94446
5 of 10 inputs records deduplicated | found 1 of 5 max possible matches | time elapsed 1.16089
6 of 10 inputs records deduplicated | found 1 of 5 max possible matches | time elapsed 1.37602
7 of 10 inputs records deduplicated | found 1 of 5 max possible matches | time elapsed 1.66245
8 of 10 inputs records deduplicated | found 1 of 5 max possible matches | time elapsed 1.98779
9 of 10 inputs records deduplicated | found 1 of 5 max possible matches | time elapsed 2.21332
10 of 10 inputs records deduplicated | found 1 of 5 max possible matches | time elapsed 2.4639
finished work at 2018-02-06 09:41:17.937386
```

```
Out[41]:                                ix_source  \
         10  31ea5edd-2c80-40ca-85ca-b6768a941d1e
         0   c204c7b7-66fd-4e55-99a4-abf647dd6b3c
         5   ff973ba5-ab42-42e0-8244-6aa82de46691

                                        ix_target  y_proba              name_source  \
         10  7ff2b1d2-bf47-4b26-849a-0d150fca7b66    1.000                  h media
         0   7a17d0c9-5992-43db-9580-c1c6cf16cdbc    0.187  l amphitryon restaurant
         5   c3200b89-b646-4b61-affb-76023e5915ef    1.000                      acm

              name_target country_code_source country_code_target      street_source  \
         10       h media                  BE                  BE      329 heistraat
         0   alter ego 31                  FR                  FR  chemin de gramont
         5            acm                  FR                  FR   9 rue de la gare

                street_target duns_source duns_target  name_fuzzyscore  \
         10      329 heistraat   372377817        None             1.00
         0   chemin de gramont   779097252   779097252             0.29
         5    9 rue de la gare   380071407        None             1.00

             street_fuzzyscore  avg_fuzzyscore  duns_exactscore  n_exactmatches
```

10	1.0	1.000	NaN	0.0
0	1.0	0.645	1.0	1.0
5	1.0	1.000	NaN	0.0

```
In [42]: df.to_excel('supervised2.xlsx')
```

3 Rule-based deduplication

it works the same as above, but instead of having to train a model, you hard-code some rules

```
In [43]: hard_threshold = {'name_tokenscore': 0.7,
                           'street_tokenscore': 0.7}
         hard_cols = list(hard_threshold.keys())

         def hardcodedfunc(r):
             r = r.fillna(0)
             for k in hard_cols:
                 if r[k] > hard_threshold[k]:
                     return 1
             else:
                 return 1

         rule_based_model = dep.FuncEvaluationModel(used_cols=hard_cols,
                                         eval_func=hardcodedfunc)
         sur=dep.Suricate(input_records=df_input_records,
                         target_records=df_target_records,
                          filterdict=filterdict,
                          intermediate_thresholds=intermediate_thresholds,
                         model=rule_based_model)
```