ipprl_tools Tutorial Notebook

August 26, 2019

0.1 ipprl_tools Tutorial Notebook

This notebook is a walk-through of the following topics: 1. Reading data using Pandas. 2. Using Synthetic Data Generation Methods. 3. Calculating Linkability Metrics on Generated Data. 4. Writing data and metric information to file.

```
In [1]: import pandas as pd
        import numpy as np
        from ipprl_tools import synthetic,metrics
        from ipprl_tools.utils import data
```

0.2 1. Reading Data Using Pandas

The module comes with a link to some pre-made synthetic data to demonstrate the corruption methods. To download it, we can use the `get_data()` method from the `utils.data` package.

```
In [2]: path = data.get_data()
```

This gets us the path to the data that has been pre-downloaded. To read in the data, we use the `read_pickle()` method from `pandas`. We use this method because it can handle reading compressed ZIP files. If your data is in CSV format, you can also use `pandas.read_csv()` to read your data in.

In either case, the `data` variable will contain a Pandas DataFrame object after calling.

Important Note: In order for the corruption methods to work correctly, the DataFrame you use must be entirely of type `np.str`. The corruption methods expect to operate on strings, and many will break on non-string data. One easy way to make sure your DataFrame is of type `np.str` is to call the function `.astype(np.str)` when reading your data. This will cast all columns of the DataFrame to be of the correct type.

We can also print out a sample of the data using `<DataFrame>.head(<num_rows>)`

```
In [3]: raw_data = pd.read_pickle(path).astype(np.str)
        # Drop some of the unecessary columns in our dataset.
        raw_data = raw_data.drop(["first_name","first_name2","last_name","last_name2","email","
        # Rename the columns of our dataset.
        raw_data.columns = ["first_name","last_name","email","address","ssn","sex","city","zip
        # Split the data into a dataset, and a swap set. We do this so that we can utilize the
        dataset = raw_data.iloc[:400000]
        swap_set = raw_data.iloc[400000:]

        dataset.head(5)
```

```
Out[3]:   first_name     last_name                    email  \
        0  Isabelita     Dommersen      idommersen0@webs.com
        1      Byrom   Le Moucheux  blemoucheux1@cornell.edu
        2     Garwin  Ismirnioglou   gismirnioglou2@army.mil
        3       Ewan        Paquet        epaquet3@baidu.com
        4     Kamila       Tailour      ktailour4@rediff.com

                               address          ssn sex         city    zip    state  \
        0                 48 Grover Way  105-17-1874   F      Houston  77281    Texas
        1            158 Marquette Hill  188-46-4510   M  Gainesville  30506  Georgia
        2  9538 Lighthouse Bay Circle  845-48-4845   M   South Bend  46620  Indiana
        3                0123 Dawn Park  886-78-7800   M      Cumming  30130  Georgia
        4              8 Linden Terrace  617-90-0336   F    Pensacola  32595  Florida

                  dob         phone        phone2        phone3          race  \
        0  2017/10/24  713-816-8206  651-608-1749  561-717-5270    Sri Lankan
        1  2017/09/10  404-582-9658  502-478-1240  540-141-9416     Colville
        2  2017/07/22  574-885-2620  626-605-9078  406-221-1811  Asian Indian
        3  2017/05/26  706-761-4259  212-881-3527  502-205-2203      Honduran
        4  2017/09/22  850-315-6220  605-784-3270  704-410-3803        Eskimo

              pcp_npi suffix      title
        0  76-5006664     Jr  Honorable
        1  49-7957492     Sr  Honorable
        2  68-4856593     Jr  Honorable
        3  78-9072361     Jr         Mr
        4  95-6884148     II         Mr
```

0.3 2. Using Synthetic Data Generation Methods

Once the data is read in, we want to apply some corruption methods on it.

```
In [4]: # We make a copy of the first few rows of data here so that we can compare it to the n
        data_to_corrupt = dataset.iloc[:5].copy()
        # The indicators dictionary will hold some information about the corruptions as they a
        indicators = {}
```

In this example, we call the `drop_per_column()` method on our small amount of sample data. We pass the function: 1. `data` - The DataFrame holding our data. 2. `indicators` - A dictionary to hold some metadata about the corruptions. 3. `columns` - We pass `columns = None` to signify that we want this operation to run on *all* columns in the DataFrame. 4. `drop_pct` - This parameter tells the function what percentage of the rows should be dropped. In our case, we want to drop 50%.

```
In [5]: synthetic.drop_per_column(data=data_to_corrupt,indicators=indicators,columns=None,drop_
```

If we compare the original results to our corrupted version, we can see the the function has randomly deleted some elements of each row (The function rounded down from 50% to 2 rows).

```
In [6]: comparison = data_to_corrupt.join(dataset.iloc[:5],lsuffix="_corrupt")
        comparison[["first_name","first_name_corrupt","last_name","last_name_corrupt","address"
```

```
Out[6]:   first_name first_name_corrupt     last_name last_name_corrupt  \
        0  Isabelita          Isabelita     Dommersen         Dommersen
        1      Byrom                      Le Moucheux
        2     Garwin             Garwin  Ismirnioglou      Ismirnioglou
        3       Ewan                           Paquet            Paquet
        4     Kamila             Kamila       Tailour

                             address     address_corrupt
        0                48 Grover Way
        1          158 Marquette Hill  158 Marquette Hill
        2  9538 Lighthouse Bay Circle
        3               0123 Dawn Park      0123 Dawn Park
        4             8 Linden Terrace    8 Linden Terrace
```

The indicators dictionary also contains information about which elements specifically were removed.

```
In [7]: def get_metrics_row(metadata, row,num_columns):
            return [None if metadata.get((i,row)) is None else metadata.get((i,row)).keys() fo

        def make_df_from_metadata(metadata,data):
            num_columns = len(data.columns)

            metrics_df = pd.DataFrame.from_dict({idx : get_metrics_row(metadata,idx,num_column
            metrics_df["type"] = "metadata"

            tmp_data = data.copy()
            tmp_data["type"] = "data"


            visual_df = pd.concat([tmp_data,metrics_df]).set_index("type",append=True).sort_in
            return visual_df
```

If we use the above helper functions above, we can view the corrupted data and the indicator metadata side-by-side. The indicator metadata records the corruptions, and in the case of more complex corruption methods, information about the corruption that was performed on each element of the synthetic dataset.

```
In [8]: meta_df = make_df_from_metadata(indicators,data_to_corrupt)
        meta_df

Out[8]:                    first_name          last_name                    email  \
          type
        0 data              Isabelita          Dommersen     idommersen0@webs.com
          metadata               None               None                     None
        1 data
          metadata  (drop_per_column)  (drop_per_column)        (drop_per_column)
        2 data                 Garwin       Ismirnioglou  gismirnioglou2@army.mil
          metadata               None               None                     None
```

```
3 data                                  Paquet
  metadata  (drop_per_column)             None        (drop_per_column)
4 data                 Kamila                       ktailour4@rediff.com
  metadata               None  (drop_per_column)                    None

                        address                ssn                sex  \
  type
0 data                                 105-17-1874                  F
  metadata    (drop_per_column)               None               None
1 data      158 Marquette Hill         188-46-4510                  M
  metadata                 None               None               None
2 data                                 845-48-4845                  M
  metadata    (drop_per_column)               None               None
3 data          0123 Dawn Park
  metadata                 None  (drop_per_column)  (drop_per_column)
4 data        8 Linden Terrace
  metadata                 None  (drop_per_column)  (drop_per_column)

                       city                zip              state  \
  type
0 data              Houston              77281              Texas
  metadata             None               None               None
1 data
  metadata  (drop_per_column)  (drop_per_column)  (drop_per_column)
2 data                                   46620            Indiana
  metadata  (drop_per_column)             None               None
3 data              Cumming              30130            Georgia
  metadata             None               None               None
4 data            Pensacola
  metadata             None  (drop_per_column)  (drop_per_column)

                        dob              phone             phone2  \
  type
0 data                                               651-608-1749
  metadata  (drop_per_column)  (drop_per_column)               None
1 data           2017/09/10       404-582-9658
  metadata             None               None  (drop_per_column)
2 data           2017/07/22       574-885-2620
  metadata             None               None  (drop_per_column)
3 data           2017/05/26       706-761-4259       212-881-3527
  metadata             None               None               None
4 data                                               605-784-3270
  metadata  (drop_per_column)  (drop_per_column)               None

                     phone3               race            pcp_npi  \
  type
0 data         561-717-5270                            76-5006664
  metadata             None  (drop_per_column)               None
```

```
1 data                                                           49-7957492
  metadata  (drop_per_column)  (drop_per_column)                      None
2 data           406-221-1811       Asian Indian                 68-4856593
  metadata               None               None                      None
3 data                                  Honduran
  metadata  (drop_per_column)               None  (drop_per_column)
4 data           704-410-3803             Eskimo
  metadata               None               None  (drop_per_column)

                       suffix              title
  type
0 data                     Jr          Honorable
  metadata               None               None
1 data
  metadata  (drop_per_column)  (drop_per_column)
2 data                                 Honorable
  metadata  (drop_per_column)               None
3 data                     Jr
  metadata               None  (drop_per_column)
4 data                     II                 Mr
  metadata               None               None
```

0.4 2.1 Chaining Synthetic Methods

To generate a synthetic dataset suitable for linkage, we can call multiple synthetic data methods, one after another, on the same data. The end result of this chain is a dataset where multiple corruptions have been performed.

```
In [9]: data_to_corrupt_large = dataset.iloc[:50].copy()
        indicators_large = {}
```

In the below code, we chain together multiples calls to synthetic methods, passing the same data and indicator variables to each method. After calling the methods, we can print out the metadata DataFrame to see which corruptions were performed for each variable value.

```
In [10]: insrt_columns = ["first_name","last_name","email"]
         insrt_freqs = [0.2,0.2,0.5]
         insrt_nums = [2,2,4]
         synthetic.string_insert_alpha(data=data_to_corrupt_large,
                                       indicators=indicators_large,
                                       insrt_num=insrt_nums,
                                       insrt_freq=insrt_freqs,
                                       columns=insrt_columns)


         n_insrt_columns = ["phone","ssn"]
         n_insrt_freqs = [0.1,0.2]
         n_insrt_nums = [2,2]
         synthetic.string_insert_numeric(data=data_to_corrupt_large,
                                         indicators=indicators_large,
```

```
                                    insrt_num=n_insrt_nums,
                                    insrt_freq=n_insrt_freqs,
                                    columns=n_insrt_columns)

drop_cols = ["first_name","last_name","email","phone","ssn"]
drop_freqs = [0.2,0.1,0.5,0.4,0.1]
synthetic.drop_per_column(data=data_to_corrupt_large,
                          indicators=indicators_large,
                          columns=drop_cols,
                          drop_pct=drop_freqs)
```

In [11]:
```
large_meta_df = make_df_from_metadata(indicators_large,data_to_corrupt_large)
large_meta_df.head(10)
```

Out[11]:
```
            first_name              last_name  \
  type
0 data        Isabelita
  metadata         None      (drop_per_column)
1 data            Byrom            Le Moucheux
  metadata         None                   None
2 data           Garwin          Ismirnioglnoua
  metadata         None  (string_insert_alpha)
3 data             Ewan                 Paquet
  metadata         None                   None
4 data           Kamila                Tailour
  metadata         None                   None

                                            email  \
  type
0 data
  metadata                      (drop_per_column)
1 data               bklemloucheux1@conrnell.evdu
  metadata                  (string_insert_alpha)
2 data                gismirnioglou2@aermby.minl
  metadata                  (string_insert_alpha)
3 data
  metadata                      (drop_per_column)
4 data
  metadata  (string_insert_alpha, drop_per_column)

                              address          ssn   sex         city  \
  type
0 data                 48 Grover Way  105-17-1874     F      Houston
  metadata                       None         None  None         None
1 data           158 Marquette Hill  188-46-4510     M  Gainesville
  metadata                       None         None  None         None
2 data     9538 Lighthouse Bay Circle  845-48-4845     M   South Bend
  metadata                       None         None  None         None
```

```
3 data                0123 Dawn Park         886-78-7800     M      Cumming
  metadata                       None                None  None         None
4 data              8 Linden Terrace                        F    Pensacola
  metadata                       None  (drop_per_column)  None         None

                  zip    state        dob              phone        phone2  \
  type
0 data          77281    Texas 2017/10/24                       651-608-1749
  metadata       None     None       None  (drop_per_column)            None
1 data          30506  Georgia 2017/09/10       404-582-9658  502-478-1240
  metadata       None     None       None               None          None
2 data          46620  Indiana 2017/07/22       574-885-2620  626-605-9078
  metadata       None     None       None               None          None
3 data          30130  Georgia 2017/05/26       706-761-4259  212-881-3527
  metadata       None     None       None               None          None
4 data          32595  Florida 2017/09/22                       605-784-3270
  metadata       None     None       None  (drop_per_column)            None

                    phone3          race     pcp_npi suffix      title
  type
0 data        561-717-5270    Sri Lankan  76-5006664     Jr  Honorable
  metadata            None          None        None   None       None
1 data        540-141-9416      Colville  49-7957492     Sr  Honorable
  metadata            None          None        None   None       None
2 data        406-221-1811  Asian Indian  68-4856593     Jr  Honorable
  metadata            None          None        None   None       None
3 data        502-205-2203      Honduran  78-9072361     Jr         Mr
  metadata            None          None        None   None       None
4 data        704-410-3803        Eskimo  95-6884148     II         Mr
  metadata            None          None        None   None       None
```

We can save this information by writing it to an Excel file using the following command.

```
In [12]: large_meta_df.to_excel("test_excel.xlsx")
```

0.5 3.1 Calculating Linkability Metrics

Once we have a dataset that has been sufficiently corrupted, we may want to calculate linkability measures on the data, to determine which columns we should use for linkage.

We can calculate metrics on the data using the `metrics` submodule.

```
In [13]: metrics.run_metrics(data_to_corrupt_large)
```

```
Out[13]:             mdr   dvr   mean_gs    std_gs  max_gs  min_gs   entropy  \
         first_name  0.2  0.82  1.000000  0.000000       1       1  4.979471
         last_name   0.1  0.92  1.000000  0.000000       1       1  5.411663
         email       0.5  0.52  1.000000  0.000000       1       1  3.321928
         address     0.0  1.00  1.000000  0.000000       1       1  5.643856
         ssn         0.1  0.92  1.000000  0.000000       1       1  5.411663
```

sex	0.0	0.04	25.000000	4.000000	29	21	0.981454
city	0.0	0.90	1.111111	0.433191	3	1	5.413661
zip	0.0	1.00	1.000000	0.000000	1	1	5.643856
state	0.0	0.44	2.272727	1.710444	7	1	4.112949
dob	0.0	0.88	1.136364	0.343174	2	1	5.403856
phone	0.4	0.62	1.000000	0.000000	1	1	3.915085
phone2	0.0	1.00	1.000000	0.000000	1	1	5.643856
phone3	0.0	1.00	1.000000	0.000000	1	1	5.643856
race	0.0	0.80	1.250000	0.487340	3	1	5.228758
pcp_npi	0.0	1.00	1.000000	0.000000	1	1	5.643856
suffix	0.0	0.10	10.000000	2.756810	13	6	2.265046
title	0.0	0.12	8.333333	3.399346	13	2	2.439941

	ptme	atf
first_name	92.943019	1.250000
last_name	97.974159	1.111111
email	70.672709	2.000000
address	100.000000	1.000000
ssn	97.974159	1.111111
sex	98.145390	25.000000
city	98.576211	1.111111
zip	100.000000	1.000000
state	92.230351	2.272727
dob	98.982029	1.136364
phone	79.025632	1.666667
phone2	100.000000	1.000000
phone3	100.000000	1.000000
race	98.249325	1.250000
pcp_npi	100.000000	1.000000
suffix	97.550239	10.000000
title	94.389800	8.333333

Each row in the above DataFrame represents a column from the original dataset. The columns in the DataFrame are various Linkability Measures, which are calculated directly from the data. For more information about what these linkability measures mean, visit this page.

0.6 4.1 Preparing Files for Linkage

To generate a dataset that we can use for linkage testing, we can use another function from the `utils.data` submodule.

In this example, we are now operating on `data`, which is the complete tutorial dataset we read in at the start of the notebook.

```
In [14]: left_ds,right_ds,gt_labels = data.split_dataset(dataset,overlap_pct=0.2)
```

In the above line of code, we used the `split_dataset` function from `ipprl_tools.utils.data` to split the dataset for us. This function accepts a set of data and splits it into two datasets, each of which has some unique rows, and some rows that overlap with the other dataset. The exact amount of overlap is configurable with the `overlap_pct` parameter.

In this case, we chose to have 20% of the rows from `dataset` appear in both `left_ds` and `right_ds`.

In addition to returning the two dataset variables, the function also returns a set of ground truth labels, `gt_labels`, which provide the `ID`s of the overlapping rows in `left_ds` and `right_ds`. If desired, you can evalaute the performance of your linkage using these known ground-truth labels.

0.6.1 4.1.1 Applying Corruption Methods

Like in Section 3, we will now apply corruption methods to the synthetic data.

This time, we must operate on two datasets, `left_ds` and `right_ds`.

Note: These methods might take a long time to run, because they are operating on very large data. If you'd like them to finish quicker, you can pass a subset of the data (using the `.iloc` function of DataFrame) to the `split_dataset()` function above to make these operations complete quicker.

```
In [15]: left_meta = {}
         synthetic.string_transpose(left_ds,left_meta,4,0.05)
         print("Transpose Complete.")
         synthetic.string_delete(left_ds,left_meta,3,0.05)
         print("Delete Complete.")
         synthetic.string_insert_alpha(left_ds,left_meta,3,0.05,columns=["first_name","last_nar
         print("Insert Alpha Complete.")
         synthetic.string_insert_numeric(left_ds,left_meta,3,0.05,columns=["phone","phone2","pl
         print("Insert Numeric Complete.")
         synthetic.edit_values(left_ds,swap_set,left_meta,0.1)
         print("Edit Values Complete.")

         columns = ["first_name",
                    "last_name",
                    "email",
                    "address",
                    "ssn",
                    "sex",
                    "city",
                    "zip",
                    "state",
                    "dob",
                    "phone",
                    "phone2",
                    "phone3",
                    "race",
                    "pcp_npi",
                    "suffix",
                    "title"]

         drop_pcts = [0.03,
                      0.03,
```

```
                    0.75,
                    0.06,
                    0.25,
                    0.07,
                    0.07,
                    0.07,
                    0.02,
                    0.02,
                    0.85,
                    0.85,
                    0.2,
                    0.2,
                    0.99,
                    0.2]

        synthetic.drop_per_column(left_ds,left_meta,drop_pct=drop_pcts,columns=columns)
        print("Per-Column Drop Complete.")
```

```
Transpose Complete.
Delete Complete.
Insert Alpha Complete.
Insert Numeric Complete.
Edit Values Complete.
Per-Column Drop Complete.
```

```
In [16]: right_meta = {}
         synthetic.string_transpose(right_ds,right_meta,4,0.05)
         print("Transpose Complete.")
         synthetic.string_delete(right_ds,right_meta,3,0.05)
         print("Delete Complete.")
         synthetic.string_insert_alpha(right_ds,right_meta,3,0.05,columns=["first_name","last_
         print("Insert Alpha Complete.")
         synthetic.string_insert_numeric(right_ds,right_meta,3,0.05,columns=["phone","phone2","
         print("Insert Numeric Complete.")
         synthetic.edit_values(right_ds,swap_set,right_meta,0.1)
         print("Edit Values Complete.")

         columns = ["first_name",
                    "last_name",
                    "email",
                    "address",
                    "ssn",
                    "sex",
                    "city",
                    "zip",
                    "state",
                    "dob",
```

```
                    "phone",
                    "phone2",
                    "phone3",
                    "race",
                    "pcp_npi",
                    "suffix",
                    "title"]

         r_drop_pcts = [0.05,
                        0.03,
                        0.75,
                        0.06,
                        0.25,
                        0.07,
                        0.07,
                        0.07,
                        0.02,
                        0.02,
                        0.80,
                        0.80,
                        0.2,
                        0.2,
                        0.99,
                        0.2]

         synthetic.drop_per_column(right_ds,right_meta,drop_pct=r_drop_pcts,columns=columns)
         print("Per-Column Drop Complete.")
```

```
Transpose Complete.
Delete Complete.
Insert Alpha Complete.
Insert Numeric Complete.
Edit Values Complete.
Per-Column Drop Complete.
```

To verify that the corruption ran on both datasets, we can run the linkability metrics on both.

```
In [17]: metrics.run_metrics(left_ds)
```

Out[17]:

	mdr	dvr	mean_gs	std_gs	max_gs	min_gs	\
first_name	0.03	0.121121	8.008807	11.659186	70	1	
last_name	0.03	0.290033	3.344491	2.410519	31	1	
email	0.75	0.249242	1.003059	0.056424	3	1	
address	0.06	0.898183	1.046562	0.237388	6	1	
ssn	0.25	0.743487	1.008765	0.099210	4	1	
sex	0.07	0.000013	111600.000000	366.000000	111966	111234	
city	0.07	0.065821	14.130160	126.591573	6386	1	
zip	0.07	0.070850	13.127095	33.471840	144	1	

state	0.02	0.007942	123.464567	1092.886732	24028	1
dob	0.02	0.033317	29.418386	126.522559	683	1
phone	0.85	0.149692	1.002088	0.046249	3	1
phone2	0.85	0.149742	1.001753	0.043143	3	1
phone3	0.20	0.792846	1.009029	0.100516	4	1
race	0.20	0.014179	56.437390	317.057947	2055	1
pcp_npi	0.99	0.010000	1.000417	0.020412	2	1
suffix	0.20	0.000025	38400.000000	509.413388	38956	37480
title	0.00	0.023921	41.804564	1218.045275	38746	1

	entropy	ptme	atf
first_name	13.341810	89.982027	8.256502
last_name	15.450870	96.045898	3.447929
email	4.777909	30.109789	4.012237
address	16.956304	95.702316	1.113364
ssn	13.891159	79.628058	1.345020
sex	1.295916	81.763224	120000.000000
city	8.732796	62.612525	15.193720
zip	11.141872	79.281341	14.115156
state	5.549127	50.926555	125.984252
dob	9.023693	69.600075	30.018762
phone	2.879569	19.028731	6.680585
phone2	2.879665	19.028765	6.678354
phone3	14.747872	84.091976	1.261286
race	6.401120	54.558452	70.546737
pcp_npi	0.193073	1.719442	100.041684
suffix	2.579369	99.783599	48000.000000
title	3.395550	27.192494	41.804564

In [18]:
```
metrics.run_metrics(right_ds)
```

Out[18]:

	mdr	dvr	mean_gs	std_gs	max_gs	min_gs \
first_name	0.05	0.118633	8.008149	11.521909	70	1
last_name	0.03	0.289067	3.355676	2.410262	26	1
email	0.75	0.249312	1.002774	0.053544	3	1
address	0.06	0.898300	1.046426	0.237438	6	1
ssn	0.25	0.743425	1.008850	0.099911	4	1
sex	0.07	0.000013	111600.000000	146.000000	111746	111454
city	0.07	0.066067	14.077578	125.705973	6274	1
zip	0.07	0.071267	13.050342	33.369686	139	1
state	0.02	0.008004	122.500000	1089.306058	24233	1
dob	0.02	0.032537	30.122951	128.012787	692	1
phone	0.80	0.199617	1.001941	0.044489	3	1
phone2	0.80	0.199492	1.002569	0.051440	3	1
phone3	0.20	0.792704	1.009209	0.101756	4	1
race	0.20	0.014033	57.024057	318.696394	2046	1
pcp_npi	0.99	0.010000	1.000417	0.020412	2	1
suffix	0.20	0.000025	38400.000000	435.762780	38826	37566

```
title       0.00  0.023758      42.090495  1222.884804   38515       1

              entropy       ptme            atf
first_name  13.153472  88.891291       8.429630
last_name   15.448238  96.058297       3.459459
email        4.778054  30.109924       4.011097
address     16.956501  95.702421       1.113219
ssn         13.891011  79.627767       1.345133
sex          1.295923  81.763607  120000.000000
city         8.745172  62.677086      15.137181
zip         11.146468  79.266327      14.032626
state        5.553737  50.916015     125.000000
dob          9.017977  69.739625      30.737705
phone        3.831299  24.641783       5.009706
phone2       3.831046  24.641588       5.012845
phone3      14.747563  84.091452       1.261511
race         6.398644  54.606761      71.280071
pcp_npi      0.193073   1.719442     100.041684
suffix       2.579396  99.784655   48000.000000
title        3.388980  27.161265      42.090495
```

We can also look at the first few rows of the data.

```
In [19]: left_ds.head()
```

```
Out[19]:    first_name   last_name                     email  \
id
0       Chaddy     Wooller
1      Adriano   Di Angelo
2        Lyell  Martinuzzi  lmartinuzzijnh@adobe.com
3      Forster   Risbrough
4    Patrizius     Hegerty

                        address          ssn sex         city    zip  \
id
0                  39 Randy Hill  413-19-0709   M      Buffalo  14269
1                3 Hagan Circle  776-75-9488   M        Miami  33190
2               44558 Cody Hill                M  Albuquerque  87110
3    24573 Messerschmidt Drive                M  Albuquerque  87180
4                  8 Buhler Park  737-25-5721   M               89012

          state         dob phone        phone2        phone3  \
id
0      New York  2018/02/26                       405-411-8832
1       Florida  2017/07/09        208-828-1705  540-633-1716
2    New Mexico  2017/04/18                       504-497-1949
3    New Mexico  2017/05/18
4       Florida  2017/11/25                       626-372-7830
```

```
                               race pcp_npi suffix      title
id
0                          Spaniard                Sr          Dr
1                           Chilean                     Honorable
2   Dominican (Dominican Republic)                             Mr
3                     Alaska Native                Sr          Mr
4                             White               III         Mrs
```

```
In [20]: right_ds.head()
```

```
Out[20]:        first_name   last_name                     email               address  \
        id
        240000     Jo-ann     Wooller                                        39 Randyill
        240001     Krisha   Di Angelo                                    8 Corscot Street
        240002      Lyell  Martinuzzi                                     44558 Cody Hill
        240003    Forster   Ribsorugh                                 0055 Mitchell Center
        240004  Patrizius     Hegerty  phegerty1g2v@google.de           8 Buhler Park

                        ssn sex         city      zip       state         dob  \
        id
        240000                M      Buffalo    14269    New York  2018/02/26
        240001  776-75-9488   M        Miami    33190     Florida  2017/07/09
        240002  634-14-4821   M  Albuquerque    87110  New Mexico  2017/04/18
        240003  108-75-5942   M  Albuquerque                       2017/05/18
        240004  737-25-5721   M    Henderson  8910112       Texas  2017/11/25

                       phone phone2        phone3                            race  \
        id
        240000                        763-391-7496                        Spaniard
        240001                        540-633-1716                          Lumbee
        240002                        504-497-1949  Dominican (Dominican Republic)
        240003  50-576-98994          402-383-2415                     Bangladeshi
        240004                        626-372-7830

               pcp_npi suffix      title
        id
        240000             Sr         Dr
        240001             Jr  Honorable
        240002            III        Rev
        240003             Sr         Mr
        240004                       Mrs
```

We can now combine these two datasets into a single dataset in order to use it as input for linkage.

```
In [21]: full_ds = pd.concat([left_ds,right_ds])
```

```
In [22]: full_ds
```

```
Out[22]:        first_name      last_name                                       email  \
id
0                   Chaddy        Wooller
1                  Adriano       Di Angelo
2                    Lyell      Martinuzzi                   lmartinuzzijnh@adobe.com
3                  Forster       Risbrough
4                Patrizius        Hegerty
5                  Gerhard       Van Halen
6                    Haily          Kydde            hquarringtondcv@macromedia.com
7                    Colly        Romanin                     chazart1ogi@mh.com.au
8                  Shepard        Ivakhin                   sivakhin16nr@oakley.com
9                  Korella          Relfe
10                   Kylen     Chanhnidng                   kchanningalf@tmall.com
11                   Gerty      Parkhouse               gparkhouse22yf@oaic.gov.au
12                   Heidi         Hrycek
13               Bette-ann        Stuckes                   bstuckes1iyw@ocn.ne.jp
14                Leontine   Peatheyjohns
15                Barnabas    Witherspoon  bwitherspoon3us@cnationalgeograephic.com
16                  Evonne  Aguirrezabala     eaguirrezabalamu1@merriam-webster.com
17                  Truman      Backshell               tbackshell2m4@youtube.com
18              Friederike        Bampton
19                    Shir           Syce            rdempster24yr@reverbnation.com
20                   Gussi         Stibbs
21                   Layla      Braitling
22                  Valery         Sidden
23                                Simmgen
24                   Judon   Hardy-Piggin
25                 Gaunnie          Hauch
26                  Arther
27                     Del        Pridham
28                 Everett           Drei
29                   Emlyn        eMwrick
...                    ...            ...                                         ...
479970             Frannie         Dragge
479971           Fernande            Udy                       fudysy5@rambler.ru
479972               Lenna          Ashbe
479973             Chelsea      Underwood              cunderwoodfgg@berkeley.edu
479974               Ogden       Shurrock
479975              Seline       Skillett
479976                Saul           Eles                     seles21lu@hhs.gov
479977             Falkner         Planke
479978             Graehme       Yantsurev
479979            Ksolrxen     Vasyutichev
479980              Leonid         Hockey
479981               Elroy          Rowan                     ecisco1854@1und1.de
479982          Gabreille          Novic                    gnovicz96@rediff.com
479983                 Con        oCgginsg                  ccoggingsty9@discuz.net
479984           Guillaume       Ferrieroi
```

479985	Reinwald	Josovitz	rgriltandpny@ucoz.ru
479986	Adrian	Troy	
479987	Cxlfiff	Cassimer	
479988	Abelard	Steutly	
479989	Bambie	Lackey	blackeynxu@ebay.co.uk
479990	Sollie	Caldero	scaledro1vzm@google.nl
479991	Emilie	Tuffs	
479992	Lowe	Kolczynski	cwalklot1x00@nps.gov
479993	Brittani	Braniff	
479994	Erhard	Hamil	ehamil53o@epa.gov
479995	Lisle	Clifft	
479996	Lorenza	Ghio	
479997	Donnie	Schutter	
479998	Rich	Learoyd	
479999			

	address	ssn	sex	city \
id				
0	39 Randy Hill	413-19-0709	M	Buffalo
1	3 Hagan Circle	776-75-9488	M	Miami
2	44558 Cody Hill		M	Albuquerque
3	24573 Messerschmidt Drive		M	Albuquerque
4	8 Buhler Park	737-25-5721	M	
5	3893 6th Point		M	Fort Lauderdale
6	7894 Rowland Plaza	732-69-1003	M	Jefferson City
7	50152 Sycamore Terrace	404-48-1735	M	Kansabs City
8	96062 Golf Point	137-99-4619	M	Tucson
9	71 4SunfeildP lace	813-46-3261	F	New York City
10	12 Quingcy Alleny		F	Jamaica
11	885 Brentwood Place	317-02-0345	F	Houston
12	7 Rockefeller Center	374-62-4187	F	Orlando
13		826-28-5433	F	hCicago
14	43509 Dovetail Park		F	Jefferson City
15	155 aLkewoodP oint	400-10-7451	M	Pueblo
16	650 Anzinger Hill		F	Washington
17	83089 Mesta Road	395-74-4581	M	Waterloo
18			F	Orlando
19		311-18-2443	F	Albuquerque
20	5517 Loftsgordon Lane	606-78-6854	F	Columbus
21	84624 Randy Circle	200-14-1268	F	Los Angeles
22	3 Longview Court	492-83-6981	F	Portland
23	31 Killdeer Junction	322-93-7379	M	
24	494 Colorado Hill	621-57-2120	M	Washintgon
25	859 Morningstar Place	637-92-3490	M	Milwaukee
26	21805 Lotheville Court	573-17-1352	M	Saint Petersburg
27	3 Killdeer Circlne		M	Myrtle Beach
28	1 Portagek Pilacec		M	Evansville
29	7Nelson Crossing		M	Virginia Beach

```
...                                ...          ...  ..                ...
479970              78747 Bay Terrace  582-32-4629   M     Daytona Beach
479971             00461 Farwell Trail  228-23-1988   F     New York City
479972  06639 Clyde Gallagher Junction  225-91-0654   F            Dallas
479973                 21 Luster Lane  807-37-3657   F         Milwaukee
479974                   8 West Drive  333-70-3956   M          Rockford
479975              9108 Warbler Park  691-13-9468   F           Phoenix
479976                   883 Randy Way               M              Troy
479977            0863 Artisan Terrace  622-56-2412   M           Phoenix
479978              05 Riverside Drive  594-01-5712   M          Alhambra
479979              2 Cardinal Street  715-88-7278   F          Pasadena
479980            87 John Wall Terrace               M
479981            8623 Delaware Parkway  401-85-4190   M          Billings
479982                     81 7th Way  560-23-5501   F            Boston
479983                                  847-45-5384   M        Des Moines
479984            4558 Golf Course Road               M        Louisville
479985      91743 Clyde Gallagher Hill  252-17-1399   M     San Francisco
479986             3126 Blackbird Pass  131-69-2444   F           Raleigh
479987              79338 7th Parkway  537-62-0696   M         San Diego
479988           7 Rockefeller Avenue  894-08-7024   M     Oklahoma City
479989                                  479-13-4071   F           Chicago
479990             419 Maxllard Street  562-78-9050   M
479991                                               F    Salt Lake City
479992              867 Packers Court                          Nashville
479993            0878 John Wall Place  855-81-7306   F     New York City
479994             772 Blackbird Place               M         eDs oMnies
479995             018 Ruskin Crossing  307-87-9596              Saarosta
479996             221 Bellgrove Alley  251-32-3345   M        Harrisburg
479997               357 Oneill Trail               M     New York City
479998          39162 Holmberg Junction  668-60-2947   M       Los Angeles
479999                                  532-46-2341   F       Minneapolis

          zip              state         dob        phone       phone2  \
id
0       14269           New York  2018/02/26
1       33190            Florida  2017/07/09                 208-828-1705
2       87110         New Mexico  2017/04/18
3       87180         New Mexico  2017/05/18
4       89012            Florida  2017/11/25
5       33310            Florida  2017/10/04  754-813-8556
6       24040           Virginia  2017/10/09
7       64193           Missouri   201703/19
8       85743       Pennsylvania  2017/10/08                 616-841-7225
9       10110           Michigan                             901-408-4706
10      70149           New York  2017/12/13
11      33111              Forda  2017/11/10
12      32868            Florida  2017/07/26
13      60669         California  2017/09/07  312-812-3790  937-862-6740
```

14	65110		2017/12/13		251-194-4114
15	81015	Colorado	2017/09/17		
16	20508	District of Columbia	2017/07/09	202-777-4927	
17	50706	Iowa	2017/09/12		
18	32854	Florida	2017/11/25		
19	87140	New Mexico	2017/04/06		
20	31904	Geogi	2017/11/15		
21	90040	California	2017/07/05		
22	97211	New York	2017/03/10		
23	2109	Massachusetts	2017/08/29		
24	20420	District of Columbia	2017/08/31		
25	1654	Massachusetts	2018/02/19		
26	33710	Florida	2017/10/28		
27	29579	South Carolina	2017/10/03		
28	47737	Indiana	2017/12/27		
29	23459	Virginia	2018/02/05		
...	...	...	...	...	...
479970		Florida	2017/08/26		
479971	10131	eNw oYrk	2017/11/15		
479972		Texas	2017/10/23		
479973	3285	Wisconsin	2017/12/12		
479974	61105	Illinois	2017/05/31		
479975	0055	Califori	2017/03/20	213-279-0601	806-446-8203
479976	48098	Michigan	2017/07/26		
479977	85062	Missouri	2017/11/10	205-493-9086	
479978	91841	California	2017/11/29	626-680-8741	
479979	45208	California	2017/04/05		
479980	30045	Kansas	2017/10/05		
479981	59112	Montana	2017/09/07	406-972-5572	
479982			2017/08/06		
479983	94142	California	2017/08/09		
479984	40287	District of Columbia	2018/02/15		
479985	94105	California	2018/02/03		
479986	27635	North Carolina	2017/07/25		
479987	92137	California	2017/03/27	619-654-8473	
479988	73135	Oklahoma	2017/08/19		
479989	60669	Illinois	2017/08/16		
479990	56372	Minnesota	2017/07/14		
479991	8619	New Jersy	2017/07/18		
479992	37250	Tennessee	2017/07/18		
479993	10292	New York	2017/06/08	212-806-6019	
479994		Iowa	2017/05/13		
479995	34238	Florida	2017/10/02		
479996	17216	Pennsylvania	2017/07/10		818-125-6558
479997	10292	New York	2017/11/28		
479998	90025	California	2017/12/10	901-487-0902	317-628-1707
479999	55423	Minnesota	2017/10/11	218-791-1122	712-998-8458

	phone3	race	pcp_npi	suffix \
id				
0	405-411-8832	Spaniard		Sr
1	540-633-1716	Chilean		
2	504-497-1949	Dominican (Dominican Republic)		
3		Alaska Native		Sr
4	626-372-7830	White		III
5	952-822-2360	Pima		Sr
6	646-826-7237	Pueblo		
7	303-756-1512	Crow		
8	904-603-8818	Houma		Jr
9	850-362-8304			
10				III
11	937-222-8318			
12	720-875-3239	Chickasaw		
13	904-958-0387	Asian		II
14	510-444-7915	Colombian	84-4384160	Sr
15	915-707-7406	Asian Indian		IV
16		Shoshone		III
17	707-566-9511	Malaysian		Jr
18	303-751-4916	American Indian		Jr
19	760-14-6391	Black or African American		IV
20		Salvadoran		IV
21	091-792-9727	Yaqui		II
22	775-461-1621	Houma		Jr
23	804-844-0908			II
24	954-909-0004	Yuman		
25		Thai		Sr
26	60520-7095-5654	Iroquois		
27	850-776-7327	Ottawa		
28				Jr
29	323-544-9097			III
...	...	...	...	...
479970	702-478-8080	Korean		III
479971	540-752-9624	Sri Lankan		
479972	704-731-9713			II
479973	757-655-6820			Sr
479974		Choctaw		Sr
479975	563-296-3356	Ute		
479976	843-781-1086	Sri Lankan		
479977	305-692-3923	Pakistani		Sr
479978	757-260-6491			III
479979	304-405-5975			II
479980	229-325-2742	Samoan		Jr
479981	408-786-8758	Central American		Jr
479982		Creek		IV
479983	505-237-3926	Paiktsani		Sr
479984	423-637-2722	American Indian		Sr

479985	720-629-5472	Vietnamese		Jr
479986		Pima		III
479987	21-3342-5509	Honduran		
479988	317-368-2662	Hmong		Jr
479989	218-645-2069			Sr
479990		Tongan	16-3932916	II
479991	248-263-7785	Potawatomi		II
479992	402-146-8352	hai		III
479993		Colombian		III
479994	305-572-5425	Chinese		Sr
479995	405-120-14208	White		III
479996	806-984-3923	Hmong		
479997	901-923-1867			Jr
479998	954-830-5188	Boliivan		IV
479999	312-416-9190	Japanese		II

	title
id	
0	Dr
1	Honorable
2	Mr
3	Mr
4	Mrs
5	Mrs
6	Mrs
7	Mrs
8	Rev
9	Rev
10	Rxeyvg
11	Mrs
12	Ms
13	Rev
14	Dr
15	Honorable
16	Ms
17	Dr
18	Honorable
19	Honorable
20	Rev
21	Rev
22	Rev
23	Dr
24	Mr
25	Mr
26	Dr
27	Honorable
28	Mrs
29	Mrs

```
...             ...
479970           Mr
479971           Ms
479972           Ms
479973           Dr
479974       Mwrbsm
479975          Rev
479976          Mrs
479977          Msz
479978           Dr
479979          Rev
479980           Dr
479981          Mrs
479982           Ms
479983          Rev
479984           Mr
479985    Honorable
479986           Mr
479987           Dr
479988           Dr
479989          Mrs
479990           Mr
479991           Ms
479992           Dr
479993           Dr
479994          Rev
479995          Rev
479996          Rev
479997           Ms
479998           Mr
479999          Rev

[480000 rows x 17 columns]
```

The `concat()` function will concatenate the two DataFrames into a single DataFrame along the axis. In our case, the `split_data()` utility function arranged it so that the indices of our index column `id`, are unique. If you did not use `split_data()` you'll want to make sure that you have references to the original IDs of your data so that you can evaluate the performance later.

`full_ds` is now a DataFrame which contains `left_ds` and `right_ds` stacked on top of each other (concatenated along the row dimension)

```
In [23]: full_ds
```

```
Out[23]:         first_name       last_name                                     email  \
         id
         0          Chaddy         Wooller
         1         Adriano       Di Angelo
         2           Lyell      Martinuzzi                 lmartinuzzijnh@adobe.com
```

3	Forster	Risbrough	
4	Patrizius	Hegerty	
5	Gerhard	Van Halen	
6	Haily	Kydde	hquarringtondcv@macromedia.com
7	Colly	Romanin	chazart1ogi@mh.com.au
8	Shepard	Ivakhin	sivakhin16nr@oakley.com
9	Korella	Relfe	
10	Kylen	Chanhnidng	kchanningalf@tmall.com
11	Gerty	Parkhouse	gparkhouse22yf@oaic.gov.au
12	Heidi	Hrycek	
13	Bette-ann	Stuckes	bstuckes1iyw@ocn.ne.jp
14	Leontine	Peatheyjohns	
15	Barnabas	Witherspoon	bwitherspoon3us@cnationalgeograephic.com
16	Evonne	Aguirrezabala	eaguirrezabalamu1@merriam-webster.com
17	Truman	Backshell	tbackshell2m4@youtube.com
18	Friederike	Bampton	
19	Shir	Syce	rdempster24yr@reverbnation.com
20	Gussi	Stibbs	
21	Layla	Braitling	
22	Valery	Sidden	
23		Simmgen	
24	Judon	Hardy-Piggin	
25	Gaunnie	Hauch	
26	Arther		
27	Del	Pridham	
28	Everett	Drei	
29	Emlyn	eMwrick	
...	...	...	...
479970	Frannie	Dragge	
479971	Fernande	Udy	fudysy5@rambler.ru
479972	Lenna	Ashbe	
479973	Chelsea	Underwood	cunderwoodfgg@berkeley.edu
479974	Ogden	Shurrock	
479975	Seline	Skillett	
479976	Saul	Eles	seles21lu@hhs.gov
479977	Falkner	Planke	
479978	Graehme	Yantsurev	
479979	Ksolrxen	Vasyutichev	
479980	Leonid	Hockey	
479981	Elroy	Rowan	ecisco1854@1und1.de
479982	Gabreille	Novic	gnovicz96@rediff.com
479983	Con	oCgginsg	ccoggingsty9@discuz.net
479984	Guillaume	Ferrieroi	
479985	Reinwald	Josovitz	rgriltandpny@ucoz.ru
479986	Adrian	Troy	
479987	Cxlfiff	Cassimer	
479988	Abelard	Steutly	
479989	Bambie	Lackey	blackeynxu@ebay.co.uk

```
479990      Sollie       Caldero                       scaledro1vzm@google.nl
479991      Emilie         Tuffs
479992        Lowe    Kolczynski                          cwalklot1x00@nps.gov
479993    Brittani       Braniff
479994      Erhard         Hamil                             ehamil53o@epa.gov
479995       Lisle        Clifft
479996     Lorenza          Ghio
479997      Donnie      Schutter
479998        Rich       Learoyd
479999

                               address          ssn sex               city  \
id
0                         39 Randy Hill  413-19-0709   M            Buffalo
1                       3 Hagan Circle  776-75-9488   M              Miami
2                        44558 Cody Hill                M        Albuquerque
3            24573 Messerschmidt Drive                M        Albuquerque
4                         8 Buhler Park  737-25-5721   M
5                        3893 6th Point                M   Fort Lauderdale
6                    7894 Rowland Plaza  732-69-1003   M    Jefferson City
7                50152 Sycamore Terrace  404-48-1735   M      Kansabs City
8                      96062 Golf Point  137-99-4619   M             Tucson
9                    71 4SunfeildP lace  813-46-3261   F     New York City
10                    12 Quingcy Alleny                F            Jamaica
11                  885 Brentwood Place  317-02-0345   F            Houston
12               7 Rockefeller Center  374-62-4187   F            Orlando
13                                       826-28-5433   F            hCicago
14                   43509 Dovetail Park                F    Jefferson City
15                  155 aLkewoodP oint  400-10-7451   M             Pueblo
16                    650 Anzinger Hill                F         Washington
17                    83089 Mesta Road  395-74-4581   M           Waterloo
18                                                     F            Orlando
19                                       311-18-2443   F        Albuquerque
20                 5517 Loftsgordon Lane  606-78-6854   F           Columbus
21                  84624 Randy Circle  200-14-1268   F        Los Angeles
22                    3 Longview Court  492-83-6981   F           Portland
23                 31 Killdeer Junction  322-93-7379   M
24                    494 Colorado Hill  621-57-2120   M         Washintgon
25                859 Morningstar Place  637-92-3490   M          Milwaukee
26               21805 Lotheville Court  573-17-1352   M  Saint Petersburg
27                   3 Killdeer Circlne                M      Myrtle Beach
28                  1 Portagek Pilacec                M        Evansville
29                    7Nelson Crossing                M    Virginia Beach
...                                ...          ...  ..                ...
479970               78747 Bay Terrace  582-32-4629   M     Daytona Beach
479971              00461 Farwell Trail  228-23-1988   F     New York City
479972  06639 Clyde Gallagher Junction  225-91-0654   F             Dallas
479973                   21 Luster Lane  807-37-3657   F          Milwaukee
```

479974	8 West Drive	333-70-3956	M	Rockford
479975	9108 Warbler Park	691-13-9468	F	Phoenix
479976	883 Randy Way		M	Troy
479977	0863 Artisan Terrace	622-56-2412	M	Phoenix
479978	05 Riverside Drive	594-01-5712	M	Alhambra
479979	2 Cardinal Street	715-88-7278	F	Pasadena
479980	87 John Wall Terrace		M	
479981	8623 Delaware Parkway	401-85-4190	M	Billings
479982	81 7th Way	560-23-5501	F	Boston
479983		847-45-5384	M	Des Moines
479984	4558 Golf Course Road		M	Louisville
479985	91743 Clyde Gallagher Hill	252-17-1399	M	San Francisco
479986	3126 Blackbird Pass	131-69-2444	F	Raleigh
479987	79338 7th Parkway	537-62-0696	M	San Diego
479988	7 Rockefeller Avenue	894-08-7024	M	Oklahoma City
479989		479-13-4071	F	Chicago
479990	419 Maxllard Street	562-78-9050	M	
479991			F	Salt Lake City
479992	867 Packers Court			Nashville
479993	0878 John Wall Place	855-81-7306	F	New York City
479994	772 Blackbird Place		M	eDs oMnies
479995	018 Ruskin Crossing	307-87-9596		Saarosta
479996	221 Bellgrove Alley	251-32-3345	M	Harrisburg
479997	357 Oneill Trail		M	New York City
479998	39162 Holmberg Junction	668-60-2947	M	Los Angeles
479999		532-46-2341	F	Minneapolis

	zip	state	dob	phone	phone2 \
id					
0	14269	New York	2018/02/26		
1	33190	Florida	2017/07/09		208-828-1705
2	87110	New Mexico	2017/04/18		
3	87180	New Mexico	2017/05/18		
4	89012	Florida	2017/11/25		
5	33310	Florida	2017/10/04	754-813-8556	
6	24040	Virginia	2017/10/09		
7	64193	Missouri	201703/19		
8	85743	Pennsylvania	2017/10/08		616-841-7225
9	10110	Michigan			901-408-4706
10	70149	New York	2017/12/13		
11	33111	Forda	2017/11/10		
12	32868	Florida	2017/07/26		
13	60669	California	2017/09/07	312-812-3790	937-862-6740
14	65110		2017/12/13		251-194-4114
15	81015	Colorado	2017/09/17		
16	20508	District of Columbia	2017/07/09	202-777-4927	
17	50706	Iowa	2017/09/12		
18	32854	Florida	2017/11/25		

```
19      87140            New Mexico  2017/04/06
20      31904                 Geogi  2017/11/15
21      90040            California  2017/07/05
22      97211              New York  2017/03/10
23       2109         Massachusetts  2017/08/29
24      20420  District of Columbia  2017/08/31
25       1654         Massachusetts  2018/02/19
26      33710               Florida  2017/10/28
27      29579        South Carolina  2017/10/03
28      47737               Indiana  2017/12/27
29      23459              Virginia  2018/02/05
...       ...                   ...         ...           ...           ...
479970                      Florida  2017/08/26
479971  10131              eNw oYrk  2017/11/15
479972                        Texas  2017/10/23
479973   3285             Wisconsin  2017/12/12
479974  61105              Illinois  2017/05/31
479975   0055              Califori  2017/03/20  213-279-0601  806-446-8203
479976  48098              Michigan  2017/07/26
479977  85062              Missouri  2017/11/10  205-493-9086
479978  91841            California  2017/11/29  626-680-8741
479979  45208            California  2017/04/05
479980  30045                Kansas  2017/10/05
479981  59112               Montana  2017/09/07  406-972-5572
479982                               2017/08/06
479983  94142            California  2017/08/09
479984  40287  District of Columbia  2018/02/15
479985  94105            California  2018/02/03
479986  27635        North Carolina  2017/07/25
479987  92137            California  2017/03/27  619-654-8473
479988  73135              Oklahoma  2017/08/19
479989  60669              Illinois  2017/08/16
479990  56372             Minnesota  2017/07/14
479991   8619             New Jersy  2017/07/18
479992  37250             Tennessee  2017/07/18
479993  10292              New York  2017/06/08  212-806-6019
479994                         Iowa  2017/05/13
479995  34238               Florida  2017/10/02
479996  17216          Pennsylvania  2017/07/10                818-125-6558
479997  10292              New York  2017/11/28
479998  90025            California  2017/12/10  901-487-0902  317-628-1707
479999  55423             Minnesota  2017/10/11  218-791-1122  712-998-8458

                 phone3                             race    pcp_npi suffix  \
id
0          405-411-8832                         Spaniard                Sr
1          540-633-1716                          Chilean
2          504-497-1949  Dominican (Dominican Republic)
```

3		Alaska Native		Sr
4	626-372-7830	White		III
5	952-822-2360	Pima		Sr
6	646-826-7237	Pueblo		
7	303-756-1512	Crow		
8	904-603-8818	Houma		Jr
9	850-362-8304			
10				III
11	937-222-8318			
12	720-875-3239	Chickasaw		
13	904-958-0387	Asian		II
14	510-444-7915	Colombian	84-4384160	Sr
15	915-707-7406	Asian Indian		IV
16		Shoshone		III
17	707-566-9511	Malaysian		Jr
18	303-751-4916	American Indian		Jr
19	760-14-6391	Black or African American		IV
20		Salvadoran		IV
21	091-792-9727	Yaqui		II
22	775-461-1621	Houma		Jr
23	804-844-0908			II
24	954-909-0004	Yuman		
25		Thai		Sr
26	60520-7095-5654	Iroquois		
27	850-776-7327	Ottawa		
28				Jr
29	323-544-9097			III
...	...	...	...	...
479970	702-478-8080	Korean		III
479971	540-752-9624	Sri Lankan		
479972	704-731-9713			II
479973	757-655-6820			Sr
479974		Choctaw		Sr
479975	563-296-3356	Ute		
479976	843-781-1086	Sri Lankan		
479977	305-692-3923	Pakistani		Sr
479978	757-260-6491			III
479979	304-405-5975			II
479980	229-325-2742	Samoan		Jr
479981	408-786-8758	Central American		Jr
479982		Creek		IV
479983	505-237-3926	Paiktsani		Sr
479984	423-637-2722	American Indian		Sr
479985	720-629-5472	Vietnamese		Jr
479986		Pima		III
479987	21-3342-5509	Honduran		
479988	317-368-2662	Hmong		Jr
479989	218-645-2069			Sr

479990		Tongan	16-3932916	II
479991	248-263-7785	Potawatomi		II
479992	402-146-8352	hai		III
479993		Colombian		III
479994	305-572-5425	Chinese		Sr
479995	405-120-14208	White		III
479996	806-984-3923	Hmong		
479997	901-923-1867			Jr
479998	954-830-5188	Boliivan		IV
479999	312-416-9190	Japanese		II

	title
id	
0	Dr
1	Honorable
2	Mr
3	Mr
4	Mrs
5	Mrs
6	Mrs
7	Mrs
8	Rev
9	Rev
10	Rxeyvg
11	Mrs
12	Ms
13	Rev
14	Dr
15	Honorable
16	Ms
17	Dr
18	Honorable
19	Honorable
20	Rev
21	Rev
22	Rev
23	Dr
24	Mr
25	Mr
26	Dr
27	Honorable
28	Mrs
29	Mrs
...	...
479970	Mr
479971	Ms
479972	Ms
479973	Dr

```
479974     Mwrbsm
479975        Rev
479976        Mrs
479977        Msz
479978         Dr
479979        Rev
479980         Dr
479981        Mrs
479982         Ms
479983        Rev
479984         Mr
479985  Honorable
479986         Mr
479987         Dr
479988         Dr
479989        Mrs
479990         Mr
479991         Ms
479992         Dr
479993         Dr
479994        Rev
479995        Rev
479996        Rev
479997         Ms
479998         Mr
479999        Rev

[480000 rows x 17 columns]
```

We can verify that the ground truth IDs from `split_data()` are still valid.

In [24]:
```
pair_num = 1
full_ds.loc[[gt_labels[pair_num][0],gt_labels[pair_num][1]]]
```

Out[24]:
```
       first_name  last_name email           address          ssn sex   city  \
id
1         Adriano  Di Angelo         3 Hagan Circle  776-75-9488   M  Miami
240001     Krisha  Di Angelo       8 Corscot Street  776-75-9488   M  Miami

          zip    state         dob phone        phone2        phone3     race  \
id
1       33190  Florida  2017/07/09        208-828-1705  540-633-1716  Chilean
240001  33190  Florida  2017/07/09                      540-633-1716   Lumbee

       pcp_npi suffix      title
id
1                      Honorable
240001             Jr  Honorable
```

Now we simply call `.to_csv()` to save our new dataset.

```
In [25]: full_ds.to_csv("test_dataset.csv")
```

In order to evaluate performance later, it is also a good idea to save the individual meta objects as well as the ground-truth labels.

```
In [26]: import pickle
         # We can save the metadata files as .pkl files, which are a common binary format for
         pickle.dump(left_meta,open("left_meta.pkl","wb"))
         pickle.dump(right_meta,open("right_meta.pkl","wb"))
         # We'll save the ground truth labels into a pikcle as well.
         pickle.dump(gt_labels,open("gt_labels.pkl","wb"))
```

To open these files again later, we can use the `pickle.load()` function in the same way we just used `pickle.dump()`

```
In [27]: test_read = pickle.load(open("gt_labels.pkl","rb"))

In [ ]:
```