ETL Pipeline Preparation

July 21, 2020

1 ETL Pipeline Preparation

Follow the instructions below to help you create your ETL pipeline. ### 1. Import libraries and load datasets. - Import Python libraries - Load `messages.csv` into a dataframe and inspect the first few lines. - Load `categories.csv` into a dataframe and inspect the first few lines.

```
In [1]: # import libraries
        import pandas as pd
        import numpy as np
        from sqlalchemy import create_engine
```

```
In [2]: # load messages dataset
        messages = pd.read_csv("messages.csv")
        messages.head()
```

```
Out[2]:    id                                            message  \
        0   2  Weather update - a cold front from Cuba that c...
        1   7            Is the Hurricane over or is it not over
        2   8                    Looking for someone but no name
        3   9  UN reports Leogane 80-90 destroyed. Only Hospi...
        4  12  says: west side of Haiti, rest of the country ...

                                                   original   genre
        0  Un front froid se retrouve sur Cuba ce matin. ...  direct
        1                 Cyclone nan fini osinon li pa fini  direct
        2  Patnm, di Maryani relem pou li banm nouvel li ...  direct
        3  UN reports Leogane 80-90 destroyed. Only Hospi...  direct
        4  facade ouest d Haiti et le reste du pays aujou...  direct
```

```
In [3]: # load categories dataset
        categories = pd.read_csv("categories.csv")
        categories.head()
```

```
Out[3]:    id                                         categories
        0   2  related-1;request-0;offer-0;aid_related-0;medi...
        1   7  related-1;request-0;offer-0;aid_related-1;medi...
        2   8  related-1;request-0;offer-0;aid_related-0;medi...
        3   9  related-1;request-1;offer-0;aid_related-1;medi...
        4  12  related-1;request-0;offer-0;aid_related-0;medi...
```

1.0.1 2. Merge datasets.

- Merge the messages and categories datasets using the common id
- Assign this combined dataset to `df`, which will be cleaned in the following steps

```
In [4]: # merge datasets
        df = messages.merge(categories, on='id')
        df.head()
```

```
Out[4]:    id                                            message  \
        0   2  Weather update - a cold front from Cuba that c...   
        1   7            Is the Hurricane over or is it not over   
        2   8                    Looking for someone but no name   
        3   9  UN reports Leogane 80-90 destroyed. Only Hospi...   
        4  12  says: west side of Haiti, rest of the country ...   

                                                    original   genre  \
        0  Un front froid se retrouve sur Cuba ce matin. ...  direct   
        1                 Cyclone nan fini osinon li pa fini  direct   
        2  Patnm, di Maryani relem pou li banm nouvel li ...  direct   
        3  UN reports Leogane 80-90 destroyed. Only Hospi...  direct   
        4  facade ouest d Haiti et le reste du pays aujou...  direct   

                                                  categories  
        0  related-1;request-0;offer-0;aid_related-0;medi...  
        1  related-1;request-0;offer-0;aid_related-1;medi...  
        2  related-1;request-0;offer-0;aid_related-0;medi...  
        3  related-1;request-1;offer-0;aid_related-1;medi...  
        4  related-1;request-0;offer-0;aid_related-0;medi...  
```

1.0.2 3. Split `categories` into separate category columns.

- Split the values in the `categories` column on the `;` character so that each value becomes a separate column. You'll find this method very helpful! Make sure to set `expand=True`.
- Use the first row of categories dataframe to create column names for the categories data.
- Rename columns of `categories` with new column names.

```
In [5]: # create a dataframe of the 36 individual category columns
        categories = categories.categories.str.split(';', expand=True)
        categories.head()
```

```
Out[5]:            0          1        2              3               4  \
        0  related-1  request-0  offer-0  aid_related-0  medical_help-0   
        1  related-1  request-0  offer-0  aid_related-1  medical_help-0   
        2  related-1  request-0  offer-0  aid_related-0  medical_help-0   
        3  related-1  request-1  offer-0  aid_related-1  medical_help-0   
        4  related-1  request-0  offer-0  aid_related-0  medical_help-0   

                           5                    6           7           8   \
0  medical_products-0  search_and_rescue-0  security-0  military-0
1  medical_products-0  search_and_rescue-0  security-0  military-0
2  medical_products-0  search_and_rescue-0  security-0  military-0
3  medical_products-1  search_and_rescue-0  security-0  military-0
4  medical_products-0  search_and_rescue-0  security-0  military-0

               9        ...             26                         27  \
0  child_alone-0       ...        aid_centers-0  other_infrastructure-0
1  child_alone-0       ...        aid_centers-0  other_infrastructure-0
2  child_alone-0       ...        aid_centers-0  other_infrastructure-0
3  child_alone-0       ...        aid_centers-0  other_infrastructure-0
4  child_alone-0       ...        aid_centers-0  other_infrastructure-0

                  28         29        30      31             32      33  \
0  weather_related-0  floods-0  storm-0  fire-0  earthquake-0  cold-0
1  weather_related-1  floods-0  storm-1  fire-0  earthquake-0  cold-0
2  weather_related-0  floods-0  storm-0  fire-0  earthquake-0  cold-0
3  weather_related-0  floods-0  storm-0  fire-0  earthquake-0  cold-0
4  weather_related-0  floods-0  storm-0  fire-0  earthquake-0  cold-0

                34                35
0  other_weather-0  direct_report-0
1  other_weather-0  direct_report-0
2  other_weather-0  direct_report-0
3  other_weather-0  direct_report-0
4  other_weather-0  direct_report-0

[5 rows x 36 columns]
```

```
In [6]: # select the first row of the categories dataframe
        row = categories.iloc[0].tolist()

        # use this row to extract a list of new column names for categories.
        # one way is to apply a lambda function that takes everything
        # up to the second to last character of each string with slicing
        category_colnames = [i.split('-')[0] for i in row]
        print(category_colnames)
```

```
['related', 'request', 'offer', 'aid_related', 'medical_help', 'medical_products', 'search_and_r
```

```
In [7]: # rename the columns of `categories`
        categories.columns = category_colnames
        categories.head()
```

```
Out[7]:      related    request    offer    aid_related    medical_help  \
        0  related-1  request-0  offer-0  aid_related-0  medical_help-0
        1  related-1  request-0  offer-0  aid_related-1  medical_help-0
        2  related-1  request-0  offer-0  aid_related-0  medical_help-0
```

```
3  related-1  request-1  offer-0  aid_related-1  medical_help-0
4  related-1  request-0  offer-0  aid_related-0  medical_help-0

     medical_products    search_and_rescue    security    military  \
0  medical_products-0  search_and_rescue-0  security-0  military-0
1  medical_products-0  search_and_rescue-0  security-0  military-0
2  medical_products-0  search_and_rescue-0  security-0  military-0
3  medical_products-1  search_and_rescue-0  security-0  military-0
4  medical_products-0  search_and_rescue-0  security-0  military-0

     child_alone       ...           aid_centers    other_infrastructure  \
0  child_alone-0       ...         aid_centers-0  other_infrastructure-0
1  child_alone-0       ...         aid_centers-0  other_infrastructure-0
2  child_alone-0       ...         aid_centers-0  other_infrastructure-0
3  child_alone-0       ...         aid_centers-0  other_infrastructure-0
4  child_alone-0       ...         aid_centers-0  other_infrastructure-0

     weather_related    floods    storm    fire    earthquake    cold  \
0  weather_related-0  floods-0  storm-0  fire-0  earthquake-0  cold-0
1  weather_related-1  floods-0  storm-1  fire-0  earthquake-0  cold-0
2  weather_related-0  floods-0  storm-0  fire-0  earthquake-0  cold-0
3  weather_related-0  floods-0  storm-0  fire-0  earthquake-0  cold-0
4  weather_related-0  floods-0  storm-0  fire-0  earthquake-0  cold-0

     other_weather    direct_report
0  other_weather-0  direct_report-0
1  other_weather-0  direct_report-0
2  other_weather-0  direct_report-0
3  other_weather-0  direct_report-0
4  other_weather-0  direct_report-0

[5 rows x 36 columns]
```

1.0.3 4. Convert category values to just numbers 0 or 1.

- Iterate through the category columns in df to keep only the last character of each string (the 1 or 0). For example, `related-0` becomes `0`, `related-1` becomes `1`. Convert the string to a numeric value.
- You can perform normal string actions on Pandas Series, like indexing, by including `.str` after the Series. You may need to first convert the Series to be of type string, which you can do with `astype(str)`.

```
In [8]: for column in categories:
            # set each value to be the last character of the string
            categories[column] = categories[column].str[-1]

            # convert column from string to numeric
            categories[column] = categories[column].astype(int)
```

```
        categories.head()
```

```
Out[8]:    related  request  offer  aid_related  medical_help  medical_products  \
        0        1        0      0            0             0                 0
        1        1        0      0            1             0                 0
        2        1        0      0            0             0                 0
        3        1        1      0            1             0                 1
        4        1        0      0            0             0                 0

           search_and_rescue  security  military  child_alone      ...        \
        0                  0         0         0            0      ...
        1                  0         0         0            0      ...
        2                  0         0         0            0      ...
        3                  0         0         0            0      ...
        4                  0         0         0            0      ...

           aid_centers  other_infrastructure  weather_related  floods  storm  fire  \
        0            0                     0                0       0      0     0
        1            0                     0                1       0      1     0
        2            0                     0                0       0      0     0
        3            0                     0                0       0      0     0
        4            0                     0                0       0      0     0

           earthquake  cold  other_weather  direct_report
        0           0     0              0              0
        1           0     0              0              0
        2           0     0              0              0
        3           0     0              0              0
        4           0     0              0              0

        [5 rows x 36 columns]
```

```
In [9]: # Look at results

        for col in categories.columns:
            if list(set(categories[col].unique().tolist()))!= [0,1]:
                print(col, categories[col].unique())

        # Replace value "2" with the most common "1" in related column
        print('Count of values in related column;')
        print(categories.related.value_counts())
        categories['related'] = categories.related.replace(2, 1)
```

```
related [1 0 2]
child_alone [0]
Count of values in related column;
1    19930
0     6125
```

```
2      193
Name: related, dtype: int64
```

1.0.4 5. Replace `categories` column in `df` with new category columns.

- Drop the categories column from the df dataframe since it is no longer needed.
- Concatenate df and categories data frames.

```
In [10]: # drop the original categories column from `df`

         df.drop('categories', axis=1, inplace=True)
         df.head()
```

```
Out[10]:    id                                            message  \
         0   2  Weather update - a cold front from Cuba that c...
         1   7            Is the Hurricane over or is it not over
         2   8                    Looking for someone but no name
         3   9  UN reports Leogane 80-90 destroyed. Only Hospi...
         4  12  says: west side of Haiti, rest of the country ...

                                                    original   genre
         0  Un front froid se retrouve sur Cuba ce matin. ...  direct
         1                 Cyclone nan fini osinon li pa fini  direct
         2  Patnm, di Maryani relem pou li banm nouvel li ...  direct
         3  UN reports Leogane 80-90 destroyed. Only Hospi...  direct
         4  facade ouest d Haiti et le reste du pays aujou...  direct
```

```
In [11]: # concatenate the original dataframe with the new `categories` dataframe
         df = pd.concat([df, categories], axis=1, join = 'inner')
         df.head()
```

```
Out[11]:    id                                            message  \
         0   2  Weather update - a cold front from Cuba that c...
         1   7            Is the Hurricane over or is it not over
         2   8                    Looking for someone but no name
         3   9  UN reports Leogane 80-90 destroyed. Only Hospi...
         4  12  says: west side of Haiti, rest of the country ...

                                                    original   genre  related  \
         0  Un front froid se retrouve sur Cuba ce matin. ...  direct        1
         1                 Cyclone nan fini osinon li pa fini  direct        1
         2  Patnm, di Maryani relem pou li banm nouvel li ...  direct        1
         3  UN reports Leogane 80-90 destroyed. Only Hospi...  direct        1
         4  facade ouest d Haiti et le reste du pays aujou...  direct        1

            request  offer  aid_related  medical_help  medical_products      ...         \
         0        0      0            0             0                 0      ...
         1        0      0            1             0                 0      ...
```

```
2        0      0            0             0                 0      ...
3        1      0            1             0                 1      ...
4        0      0            0             0                 0      ...

   aid_centers  other_infrastructure  weather_related  floods  storm  fire  \
0            0                     0                0       0      0     0
1            0                     0                1       0      1     0
2            0                     0                0       0      0     0
3            0                     0                0       0      0     0
4            0                     0                0       0      0     0

   earthquake  cold  other_weather  direct_report
0           0     0              0              0
1           0     0              0              0
2           0     0              0              0
3           0     0              0              0
4           0     0              0              0

[5 rows x 40 columns]
```

1.0.5 6. Remove duplicates.

- Check how many duplicates are in this dataset.
- Drop the duplicates.
- Confirm duplicates were removed.

```
In [12]: # check number of duplicates
         print('The number of duplicates:', df.duplicated().sum())
         print('\nThe number of duplicates i neach column:')
         df[df.duplicated()].count()
```

```
The number of duplicates: 41

The number of duplicates i neach column:


Out[12]: id                        41
         message                   41
         original                  23
         genre                     41
         related                   41
         request                   41
         offer                     41
         aid_related               41
         medical_help              41
         medical_products          41
         search_and_rescue         41
         security                  41
         military                  41
```

```
child_alone               41
water                     41
food                      41
shelter                   41
clothing                  41
money                     41
missing_people            41
refugees                  41
death                     41
other_aid                 41
infrastructure_related    41
transport                 41
buildings                 41
electricity               41
tools                     41
hospitals                 41
shops                     41
aid_centers               41
other_infrastructure      41
weather_related           41
floods                    41
storm                     41
fire                      41
earthquake                41
cold                      41
other_weather             41
direct_report             41
dtype: int64
```

```
In [13]: # drop duplicates
         df.drop_duplicates(inplace=True)
```

```
In [14]: # check number of duplicates
         df.duplicated().sum()
```

```
Out[14]: 0
```

1.0.6 7. Save the clean dataset into an sqlite database.

You can do this with pandas to_sql method combined with the SQLAlchemy library. Remember to import SQLAlchemy's `create_engine` in the first cell of this notebook to use it below.

```
In [15]: engine = create_engine('sqlite:///DisasterResponse.db')
         df.to_sql('DisasterResponse', engine, index=False, if_exists='replace')
```

1.0.7 8. Use this notebook to complete `etl_pipeline.py`

Use the template file attached in the Resources folder to write a script that runs the steps above to create a database based on new datasets specified by the user. Alternatively, you can complete `etl_pipeline.py` in the classroom on the `Project Workspace IDE` coming later.

```
In [ ]:
```