query_filter

October 25, 2017

1 Filter, Drop Nulls, Dedupe

Use `data_08.csv` and `data_18.csv`

```
In [1]: import pandas as pd
```

```
In [2]: # load datasets

        df_08 = pd.read_csv('data_08.csv')
        df_08.head(1)
```

```
Out[2]:        model  displ      cyl    trans drive      fuel cert_region veh_class  \
        0  ACURA MDX    3.7  (6 cyl)  Auto-S5   4WD  Gasoline          CA       SUV

          air_pollution_score city_mpg hwy_mpg cmb_mpg greenhouse_gas_score smartway
        0                   7       15      20      17                    4       no
```

```
In [3]: df_18 = pd.read_csv('data_18.csv')
        df_18.head(1)
```

```
Out[3]:        model  displ  cyl       trans drive      fuel cert_region  veh_class  \
        0  ACURA RDX    3.5  6.0  SemiAuto-6   2WD  Gasoline          FA  small SUV

           air_pollution_score city_mpg hwy_mpg cmb_mpg  greenhouse_gas_score smartway
        0                    3       20      28      23                     5       No
```

```
In [4]: # view dimensions of dataset
        df_08.shape
```

```
Out[4]: (2404, 14)
```

```
In [5]: # view dimensions of dataset
        df_18.shape
```

```
Out[5]: (1611, 14)
```

1.1 Filter by Certification Region

```
In [6]: # filter datasets for rows following California standards
        df_08 = df_08.query('cert_region == "CA"')
        df_18 = df_18.query('cert_region == "CA"')

In [7]: # confirm only certification region is California
        df_08['cert_region'].unique()

Out[7]: array(['CA'], dtype=object)

In [8]: # confirm only certification region is California
        df_18['cert_region'].unique()

Out[8]: array(['CA'], dtype=object)

In [9]: # drop certification region columns form both datasets
        df_08.drop(['cert_region'], axis = 1, inplace = True)
        df_18.drop(['cert_region'], axis = 1, inplace = True)

In [10]: df_08.shape

Out[10]: (1084, 13)

In [11]: df_18.shape

Out[11]: (798, 13)
```

1.2 Drop Rows with Missing Values

```
In [16]: # view missing value count for each feature in 2008
         df_08.isnull().sum()

Out[16]: model                    0
         displ                    0
         cyl                     75
         trans                   75
         drive                   37
         fuel                     0
         veh_class                0
         air_pollution_score      0
         city_mpg                75
         hwy_mpg                 75
         cmb_mpg                 75
         greenhouse_gas_score    75
         smartway                 0
         dtype: int64

In [22]: # view missing value count for each feature in 2018
         df_18.isnull().sum()
```

```
Out[22]: model                   0
         displ                   1
         cyl                     1
         trans                   0
         drive                   0
         fuel                    0
         veh_class               0
         air_pollution_score     0
         city_mpg                0
         hwy_mpg                 0
         cmb_mpg                 0
         greenhouse_gas_score    0
         smartway                0
         dtype: int64
```

```
In [23]: # drop rows with any null values in both datasets
         df_08.dropna(how='any', inplace = True);
         df_18.dropna(how='any', inplace = True);
```

```
In [24]: # checks if any of columns in 2008 have null values - should print False
         df_08.isnull().sum().any()
```

```
Out[24]: False
```

```
In [25]: # checks if any of columns in 2018 have null values - should print False
         df_18.isnull().sum().any()
```

```
Out[25]: False
```

1.3 Dedupe Data

```
In [26]: # print number of duplicates in 2008 and 2018 datasets
         print(sum(df_08.duplicated()))
         print(sum(df_18.duplicated()))
```

```
23
3
```

```
In [27]: # drop duplicates in both datasets
         df_08.drop_duplicates(inplace=True)
         df_18.drop_duplicates(inplace=True)
```

```
In [28]: # print number of duplicates again to confirm dedupe - should both be 0

         print(sum(df_08.duplicated()))
         print(sum(df_18.duplicated()))
```

```
0
0
```

```
In [29]: # save progress for the next section
         df_08.to_csv('data_08.csv', index=False)
         df_18.to_csv('data_18.csv', index=False)

In [ ]:
```