Michael Perrine

Week 7/8 Assignment

DSC 540

Professor Williams

Data Cleaning and Transformations

The goal of this assignment is to clean and transform two datasets. The first one is the candy data set and the second is the met data set. The candy dataset is survey data of candy. The met dataset is a collection of artwork held at the Metropolitan Museum of Art. With in the analysis I will perform several tasks. The tasks include Filter out missing data, Transform data using either mapping or a function, Create hierarchical index, Pivot the data, Grouping with Functions, Split/Apply/Combine, Generate date range, and Convert timestamps to periods and back.

In [ ]:
```
# This code imports the necessary libraries
import pandas as pd
import warnings
```

In [ ]:
```
# This code suppresses minor warnings
warnings.simplefilter('ignore')
```

In [ ]:
```
# This code uploads the candy dataset and displays the first five rows
candy = pd.read_excel(r'candyhierarchy2017.xlsx')
candy.head()
```

Out[ ]:

	Internal ID	Q1: GOING OUT?	Q2: GENDER	Q3: AGE	Q4: COUNTRY	Q5: STATE, PROVINCE, COUNTY, ETC	Q6 | 100 Grand Bar	Q6 | Anonymous brown globs that come in black and orange wrappers\t(a.k.a. Mary Janes)	Q A fu siz can b
0	90258773	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
1	90272821	No	Male	44	USA	NM	MEH	DESPAIR	J
2	90272829	NaN	Male	49	USA	Virginia	NaN	NaN	N
3	90272840	No	Male	40	us	or	MEH	DESPAIR	J
4	90272841	No	Male	23	usa	exton pa	JOY	DESPAIR	J

5 rows × 120 columns

In [ ]:
```
# This code imports the met data set and displays the first 5 rows
met = pd.read_csv('MetObjects.txt', sep=",")
met.head()
```

Out[ ]:

	Object Number	Is Highlight	Is Timeline Work	Is Public Domain	Object ID	Gallery Number	Department	AccessionYear
0	1979.486.1	False	False	False	1	NaN	The American Wing	1979.0
1	1980.264.5	False	False	False	2	NaN	The American Wing	1980.0
2	67.265.9	False	False	False	3	NaN	The American Wing	1967.0
3	67.265.10	False	False	False	4	NaN	The American Wing	1967.0
4	67.265.11	False	False	False	5	NaN	The American Wing	1967.0

5 rows × 54 columns

1. Filter out missing data

In [ ]:
```
# This code displays the types of data in the candy data set
candy.dtypes
```

Out[ ]:
```
Internal ID                 int64
Q1: GOING OUT?             object
Q2: GENDER                 object
Q3: AGE                    object
Q4: COUNTRY                object
                            ...
Q12: MEDIA [Daily Dish]    float64
Q12: MEDIA [Science]       float64
Q12: MEDIA [ESPN]          float64
Q12: MEDIA [Yahoo]         float64
Click Coordinates (x, y)    object
Length: 120, dtype: object
```

In [ ]:
```
# This code gives the dimension for the candy data
candy.shape
```

Out[ ]:
```
(2460, 120)
```

In [ ]:
```
# This code displays the sum of the null values
candy.isnull().sum()
```

Out[ ]:
```
Internal ID                   0
Q1: GOING OUT?              110
Q2: GENDER                   41
Q3: AGE                      84
Q4: COUNTRY                  64
                           ...
Q12: MEDIA [Daily Dish]    2375
Q12: MEDIA [Science]       1098
Q12: MEDIA [ESPN]          2361
Q12: MEDIA [Yahoo]         2393
Click Coordinates (x, y)    855
Length: 120, dtype: int64
```

In [ ]:
```
# This code fills the null values with a zero and displays the sum of any null valu
candy_1 = candy.fillna(0)
candy_1.isnull().sum()
```

Out[ ]:
```
Internal ID                 0
Q1: GOING OUT?              0
Q2: GENDER                  0
Q3: AGE                     0
Q4: COUNTRY                 0
                           ..
Q12: MEDIA [Daily Dish]     0
Q12: MEDIA [Science]        0
Q12: MEDIA [ESPN]           0
Q12: MEDIA [Yahoo]          0
Click Coordinates (x, y)    0
Length: 120, dtype: int64
```

In [ ]:
```
# This code displays the first five rows after replacing null values
candy_1.head()
```

Out[ ]:

	Internal ID	Q1: GOING OUT?	Q2: GENDER	Q3: AGE	Q4: COUNTRY	Q5: STATE, PROVINCE, COUNTY, ETC	Q6 | 100 Grand Bar	Q6 | Anonymous brown globs that come in black and orange wrappers\t(a.k.a. Mary Janes)	Q A fu siz can b
0	90258773	0	0	0	0	0	0	0	
1	90272821	No	Male	44	USA	NM	MEH	DESPAIR	J
2	90272829	0	Male	49	USA	Virginia	0	0	
3	90272840	No	Male	40	us	or	MEH	DESPAIR	J
4	90272841	No	Male	23	usa	exton pa	JOY	DESPAIR	J

5 rows × 120 columns

2. Transform data using either mapping or a function

In [ ]:
```
# This code maps to the department column and adds "This is the department of" in f
met_1 = met["Department"].map("This is the department of {}".format)
met_1.head()
```

Out[ ]:
```
0    This is the department of The American Wing
1    This is the department of The American Wing
2    This is the department of The American Wing
3    This is the department of The American Wing
4    This is the department of The American Wing
Name: Department, dtype: object
```

3. Create hierarchical index

In [ ]:
```
# this code displays the range of the candy index
candy.index
```

Out[ ]:
```
RangeIndex(start=0, stop=2460, step=1)
```

In [ ]:
```
# This code creates an index hierarchy for the candy data
candy_2 = candy_1.set_index(["Q4: COUNTRY", "Q5: STATE, PROVINCE, COUNTY, ETC", "Q2
candy_2.head()
```

Out[ ]:

	Internal ID	Q1: GOING OUT?	Q3: AGE	Q6 | 100 Grand Bar	Q6 | Anonymous brown globs that come in black and orange wrappers\t(a.k.a. Mary Janes)	Q6 | Any full-sized candy bar	Q4: COUNTRY	Q5: STATE, PROVINCE, COUNTY, ETC	Q2: GENDER
0	90258773	0	0	0	0	0	**0**	**0**	**0**
USA	90272821	No	44	MEH	DESPAIR	JOY	**USA**	**NM**	**Male**
USA	90272829	0	49	0	0	0	**USA**	**Virginia**	**Male**
us	90272840	No	40	MEH	DESPAIR	JOY	**us**	**or**	**Male**
usa	90272841	No	23	JOY	DESPAIR	JOY	**usa**	**exton pa**	**Male**

5 rows × 117 columns

4. Pivot the data

In [ ]:
```
# This code displays the first five rows of the met data
met.head()
```

Out[ ]:

	Object Number	Is Highlight	Is Timeline Work	Is Public Domain	Object ID	Gallery Number	Department	AccessionYear
0	1979.486.1	False	False	False	1	NaN	The American Wing	1979.0
1	1980.264.5	False	False	False	2	NaN	The American Wing	1980.0
2	67.265.9	False	False	False	3	NaN	The American Wing	1967.0
3	67.265.10	False	False	False	4	NaN	The American Wing	1967.0
4	67.265.11	False	False	False	5	NaN	The American Wing	1967.0

5 rows × 54 columns

In [ ]:
```
# This code pivots the met data
pivot_met = met.pivot(index='AccessionYear', columns='Object ID', values='Object Nu
pivot_met.tail()
```

Out[ ]:

Object ID	1	2	3	4	5	6	7	8	9	10	...	900494
AccessionYear												
2021	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN
2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	2022.459.13.5
2022-02-09	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN
2022-05-20	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN
2023	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN

5 rows × 484956 columns

5. Grouping with Functions

In [ ]:
```
# This code converts the accession year column from an object to a date
met['AccessionYear']= pd.to_datetime(met['AccessionYear'])
```

In [ ]:
```
# This code converts the object number column from an object to a number
met['Object Number'] = pd.to_numeric(met['Object Number'], errors='coerce')
```

In [ ]:
```
# This code creates a group by using accession year and object number and displays
met_group = met.groupby('AccessionYear')['Object Number'].mean()
met_group.head()
```

Out[ ]:
```
AccessionYear
1874-01-01   NaN
1883-01-01   NaN
1885-01-01   NaN
1886-01-01   NaN
1889-01-01   NaN
Name: Object Number, dtype: float64
```

6. Split/Apply/Combine

In [ ]:
```
# This code renames the object name column
met = met.rename(columns={'Object Name' : "Object_Name"})
```

In [ ]:
```
# this code displays the unique values in the column Object_Name
met["Object_Name"].unique
```

Out[ ]:
```
<bound method Series.unique of 0          Coin
1          Coin
2          Coin
3          Coin
4          Coin
           ...
484951    Print
484952    Print
484953    Print
484954      NaN
484955    Print
Name: Object_Name, Length: 484956, dtype: object>
```

In [ ]:
```
# This code creates a new data frame grouped by object name
met_2 = met.groupby('Object_Name')
met_2
```

Out[ ]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x0000023F0AE3C440>
```

In [ ]:
```
# This code creates a for loop to cycle through the object name in the met_2 data f
for Object_Name, Object_Name_met in met_2:
    print(Object_Name)
    print(Object_Name_met)
```

In [ ]:
```
# This code creates a data frame of the prints in the met_2 data frame
met_2.get_group('Print')
```

7. Generate date range

In [53]:
```
# This code displays the dimension of the candy_1 data frame
candy_1.shape
```

Out[53]:
```
(2460, 120)
```

In [58]:
```
# This code creates a date range for the candy_1 data set
candy_1["Date"] = pd.date_range(start="2020-01-01", periods=2460, freq="D")
candy_1.head()
```

Out[58]:

	Internal ID	Q1: GOING OUT?	Q2: GENDER	Q3: AGE	Q4: COUNTRY	Q5: STATE, PROVINCE, COUNTY, ETC	Q6 | 100 Grand Bar	Q6 | Anonymous brown globs that come in black and orange wrappers\t(a.k.a. Mary Janes)	Q A fu siz can b
0	90258773	0	0	0	0	0	0	0	
1	90272821	No	Male	44	USA	NM	MEH	DESPAIR	J
2	90272829	0	Male	49	USA	Virginia	0	0	
3	90272840	No	Male	40	us	or	MEH	DESPAIR	J
4	90272841	No	Male	23	usa	exton pa	JOY	DESPAIR	J

5 rows × 121 columns

8. Convert timestamps to periods and back

In [ ]:
```
# This converts the accession year column from an object to a time stamp
met["AccessionYear"]= pd.to_datetime(met["AccessionYear"])
```

In [ ]:
```
# This code converts the time stamp into a period and displays the result
met["Accession_Period"]= met["AccessionYear"].dt.to_period("M")
met["Accession_Period"]
```

Out[ ]:
```
0         1970-01
1         1970-01
2         1970-01
3         1970-01
4         1970-01
           ...
484951    1955-01
484952    1977-01
484953    1933-01
484954        NaT
484955    1917-01
Name: Accession_Period, Length: 484956, dtype: period[M]
```

In [ ]:
```
# This code converts the period back into a time stamp and displays the result
met["AccessionYear_converted"] = met["Accession_Period"].dt.to_timestamp()
met["AccessionYear_converted"]
```

Out[ ]:
```
0         1970-01-01
1         1970-01-01
2         1970-01-01
3         1970-01-01
4         1970-01-01
             ...
484951    1955-01-01
484952    1977-01-01
484953    1933-01-01
484954           NaT
484955    1917-01-01
Name: AccessionYear_converted, Length: 484956, dtype: datetime64[ns]
```