Bilenkin540Weeks_7_&_8_Exercises

April 24, 2025

0.1 Loading the "So Much Data Candy, Seriously" dataset

[535]:
```
# Defining the file path
import pandas as pd

file_path = r"C:\Users\maxim\OneDrive\Desktop\BU\DSC␣
 ↪540\CANDY-HIERARCHY-2015-SURVEY-Responses.xlsx"

# Loading the Excel file
df = pd.read_excel(file_path)

# Displaying information and the first two rows from dataset
print("Shape of dataset:", df.shape)
# df.head(2) # Commenting out because the output takes too many pages in PDF␣
 ↪document
```

```
Shape of dataset: (5630, 124)
```

0.1.1 Chapter 7 – Filter Out Missing Data

[536]:
```
# Displaying number of missing values per column
missing_counts = df.isnull().sum()
print("Missing values per column:\n")
print(missing_counts[missing_counts > 0])

# Droping rows where all candy ratings are missing (JOY/DESPAIR/MEH)
candy_columns = [col for col in df.columns if col.startswith('[') and col.
 ↪endswith(']')]
df_filtered = df.dropna(subset=candy_columns, how='all')

# Displaying outcomes
print(f"\nOriginal dataset shape: {df.shape}")
print(f"After filtering rows with all candy ratings missing: {df_filtered.
 ↪shape}")
```

```
Missing values per column:

How old are you?
199
```

```
 [Butterfinger]
383
 [100 Grand Bar]
656
 [Anonymous brown globs that come in black and orange wrappers]
298
 [Any full-sized candy bar]
277
                 …
Please estimate the degrees of separation you have from the following folks
[Thom Yorke]         5630
Please estimate the degrees of separation you have from the following folks [JJ
Abrams]          5630
Please estimate the degrees of separation you have from the following folks
[Hillary Clinton]    5630
Please estimate the degrees of separation you have from the following folks
[Donald Trump]       5630
Please estimate the degrees of separation you have from the following folks
[Beyoncé Knowles]    5630
Length: 122, dtype: int64

Original dataset shape: (5630, 124)
After filtering rows with all candy ratings missing: (0, 124)
```

0.2 Chapter 7 – Filter Out Missing Data

Will drop rows where **all** candy rating columns are missing and keeping once with at least one response. This helps to analyze only the relevant survey responses.

[537]:
```
# Select only columns that look like candy ratings
candy_columns = [col for col in df.columns if col.strip().startswith('[') and␣
 ↪col.strip().endswith(']')]

# Check for missing values in these candy columns
missing_per_candy_column = df[candy_columns].isnull().sum()
print("Missing values per candy column:\n", missing_per_candy_column)

# Drop rows where ALL candy columns are missing
df_filtered = df.dropna(subset=candy_columns, how='all')

print(f"\nOriginal dataset shape: {df.shape}")
print(f"After filtering rows with all candy ratings missing: {df_filtered.
 ↪shape}")

# Display 2 sample rows of the filtered candy columns
# Commenting out to save space in the PDF
# df_filtered[candy_columns].head(2)  # Commenting out because the output takes␣
 ↪too many pages in PDF document
```

```
Missing values per candy column:
 [Butterfinger]
383
[100 Grand Bar]
656
[Anonymous brown globs that come in black and orange wrappers]
298
[Any full-sized candy bar]
277
[Black Jacks]
991
           …
[White Bread]
667
[Whole Wheat anything]
683
[York Peppermint Patties]
464
[Sea-salt flavored stuff, probably chocolate, since this is the "it" flavor of
the year]    580
[Necco Wafers]
683
Length: 95, dtype: int64

Original dataset shape: (5630, 124)
After filtering rows with all candy ratings missing: (5533, 124)
```

0.3 Chapter 7 – Filter Out Missing Data

To begin cleaning the dataset, I focused on the candy rating columns (which contain values like "JOY", "MEH", and "DESPAIR"). First, I calculated how many missing values each candy column had. After, I removed any survey responses where *all* candy ratings were missing.

This step made the dataset smaller by reducing it from **5,630 rows** to **5,533 rows**, ensuring that I only keep entries with at least one valid candy rating.

0.4 Chapter 7 – Replacing Missing Values

[538]:
```python
# Replacing missing values with the most frequent value (mode) in each candy␣
 ↪column
candy_columns = [col for col in df.columns if col.startswith('[')]  #␣
 ↪Identifying candy columns

# Replacing missing values with the mode (most frequent value) of each column
for column in candy_columns:
    mode_value = df[column].mode()[0]  # Getting the most frequent value in the␣
 ↪column
```

```
    df[column].fillna(mode_value, inplace=True)  # Replacing missing values␣
↪with mode

# Verifying the changes
missing_values_after_replacement = df[candy_columns].isnull().sum()
print(f"Missing values after replacement: \n{missing_values_after_replacement}")
```

```
Missing values after replacement:
Series([], dtype: float64)
```

0.5 Chapter 7 – Replacing Missing Values

After I filtered out rows with all the missing candy ratings, the next step was to handle the remaining missing values in the candy columns. To ensure the dataset remains as complete as possible, I replaced missing values with the most frequent rating (mode) in each candy column. This approach helps maintain the integrity of the dataset without losing a lot of information.

With this step, I was able to fill in the missing values with the most common ratings for each candy, leaving no missing data in these columns.

0.5.1 Chapter 8 – Combining Data.

Method 1: Concatenation

[539]:
```
import pandas as pd

# Loading the original dataset
file_path = r"C:\Users\maxim\OneDrive\Desktop\BU\DSC␣
↪540\CANDY-HIERARCHY-2015-SURVEY-Responses.xlsx"
df = pd.read_excel(file_path)

# Displaying the first two rows of the dataset to confirm it loaded correctly
# Commenting out because the output takes too many pages in PDF document
# df.head(2)  # Commented out to reduce the number of pages in the PDF
```

[540]:
```
# Filtering the candy columns (those with names starting with a space and a␣
↪square bracket)
candy_columns = df.columns[df.columns.str.startswith(' [')]

# Extracting the relevant candy data
candy_data = df[candy_columns]

# Displaying the first row of candy data to confirm
# print(candy_data.head(1)) Commenting out print statement because it takes too␣
↪many pages on pdf document.
```

[541]:
```
# Splitting the dataset into two parts for demonstration
part1 = candy_data.iloc[:3000]
part2 = candy_data.iloc[3000:]
```

```
# Displaying the shapes of the two parts
print(f"Part 1 shape: {part1.shape}")
print(f"Part 2 shape: {part2.shape}")
```

```
Part 1 shape: (3000, 95)
Part 2 shape: (2630, 95)
```

[542]:
```
# Dropping columns with too many missing values by keeping top 15 columns with␣
↪least NaNs
na_counts = combined_candy_data.isnull().sum()
top_columns = na_counts.nsmallest(15).index  # Select columns with the fewest␣
↪NaNs
combined_candy_data_clean = combined_candy_data[top_columns]

# Displaying the shape and a few rows to confirm
print(f"Shape after keeping top 15 most complete columns:␣
↪{combined_candy_data_clean.shape}")
combined_candy_data_clean.head(2)
```

```
Shape after keeping top 15 most complete columns: (5630, 15)
```

[542]:
```
  [Reese's Peanut Butter Cups] [Any full-sized candy bar] [Snickers]  \
0                          JOY                        JOY        JOY
1                          JOY                        JOY        JOY

  [Kit Kat] [Anonymous brown globs that come in black and orange wrappers]  \
0       NaN                                            DESPAIR
1       JOY                                            DESPAIR

  [Cash, or other forms of legal tender] [Peanut M&M's]  \
0                                    JOY            JOY
1                                    JOY            JOY

  [Creepy Religious comics/Chick Tracts] [Dental paraphenalia]  \
0                                DESPAIR               DESPAIR
1                                DESPAIR               DESPAIR

  [Broken glow stick]  \
0             DESPAIR
1             DESPAIR

  [Candy that is clearly just the stuff given out for free at restaurants]  \
0                                            DESPAIR
1                                            DESPAIR

  [Twix] [Dark Chocolate Hershey] [Cadbury Creme Eggs] [Butterfinger]
0    NaN                      JOY              DESPAIR            JOY
```

```
1    JOY                        DESPAIR                  DESPAIR               JOY
```

0.6 Chapter 8 – Combining Data

In this example, I demonstrated how to combine data by first splitting the cleaned candy ratings data into two parts. I then used pd.concat() to combine these two parts back into a single DataFrame. This simulates combining two similar datasets that share the same structure (i.e., columns).

Due to the large number of columns in the combined dataset, I had to drop some columns to ensure the output would fit within the page width in the PDF. Despite dropping several columns, the final combined dataset retained the same number of rows and the remaining columns, confirming that the concatenation process was successful.

0.7 Chapter 8 – Merging Data

Method 2: Merging on Index or Column

[543]:
```
import pandas as pd

# Loading the Excel file
file_path = r"C:\Users\maxim\OneDrive\Desktop\BU\DSC␣
↪540\CANDY-HIERARCHY-2015-SURVEY-Responses.xlsx"
excel_data = pd.ExcelFile(file_path)

# Checking sheet names to identify the data available
print(excel_data.sheet_names)

# Loading the first sheet and inspecting the first few rows
df = pd.read_excel(file_path, sheet_name=excel_data.sheet_names[0])

# Inspecting the first row of the data
df.head(1) # Printing only one row to shrink output on PDF
```

```
['Form Responses 1']
```

[543]:
```
                Timestamp How old are you?  \
0 2015-10-23 08:46:20.451               35

  Are you going actually going trick or treating yourself?  [Butterfinger]  \
0                                                 No                    JOY

   [100 Grand Bar]  \
0              NaN

   [Anonymous brown globs that come in black and orange wrappers]  \
0                                            DESPAIR

   [Any full-sized candy bar]  [Black Jacks]  [Bonkers]  [Bottle Caps]  \
0                           JOY             NaN        NaN             NaN

   [Box’o’ Raisins]  [Brach products (not including candy corn)]  \
0              NaN                                     DESPAIR

   [Bubble Gum]  [Cadbury Creme Eggs]  [Candy Corn]  \
0      DESPAIR               DESPAIR           NaN

   [Vials of pure high fructose corn syrup, for main-lining into your vein]  \
0                                            DESPAIR

   [Candy that is clearly just the stuff given out for free at restaurants]  \
0                                            DESPAIR

   [Cash, or other forms of legal tender]  [Chiclets]  [Caramellos]  \
0                                    JOY         NaN           NaN

   [Snickers]  [Dark Chocolate Hershey]  [Dental paraphenalia]  [Dots]  \
0        JOY                       JOY                DESPAIR     JOY

   [Fuzzy Peaches]  [Generic Brand Acetaminophen]  [Glow sticks]  \
0         DESPAIR                        DESPAIR        DESPAIR

   [Broken glow stick]  [Goo Goo Clusters]  [Good N' Plenty]  \
0             DESPAIR                 NaN           DESPAIR

   [Gum from baseball cards]  [Gummy Bears straight up]  \
0                   DESPAIR                    DESPAIR

   [Creepy Religious comics/Chick Tracts]  [Healthy Fruit]  [Heath Bar]  \
0                               DESPAIR          DESPAIR          NaN

   [Hershey’s Kissables]  [Hershey’s Milk Chocolate]  \
0                   NaN                        JOY

   [Hugs (actual physical hugs)]  [Jolly Rancher (bad flavor)]  \
0                       DESPAIR                           NaN

   [Jolly Ranchers (good flavor)]  [Kale smoothie]  [Kinder Happy Hippo]  \
0                             NaN              NaN                   NaN

   [Kit Kat]  [Hard Candy]  [Lapel Pins]  [LemonHeads]  [Licorice]  \
0       NaN           NaN           NaN           NaN         NaN

   [Licorice (not black)]  [Lindt Truffle]  [Lollipops]  [Mars]  [Mary Janes]  \
0                     NaN              NaN          NaN     NaN           NaN
```

```
   [Maynards]  [Milk Duds]  [LaffyTaffy]  [Minibags of chips]  \
0         NaN          NaN           NaN                  NaN

   [JoyJoy (Mit Iodine)]  [Reggie Jackson Bar]  [Pixy Stix]  [Nerds]  \
0                    NaN                   NaN          NaN      NaN

   [Nestle Crunch]  [Now'n'Laters]  [Pencils]  [Milky Way]  \
0              NaN             NaN        NaN          JOY

   [Reese's Peanut Butter Cups]  [Tolberone something or other]  [Runts]  \
0                           JOY                             NaN      NaN

   [Junior Mints]  [Senior Mints]  [Mint Kisses]  [Mint Juleps]  \
0             JOY             NaN            NaN            NaN

   [Mint Leaves]  [Peanut M&M's]  [Regular M&Ms]  [Mint M&Ms]  [Ribbon candy]  \
0            NaN             JOY             JOY          NaN             NaN

   [Rolos]  [Skittles]  [Smarties (American)]  [Smarties (Commonwealth)]  \
0      JOY         NaN                    JOY                        NaN

   [Chick-o-Sticks (we don't know what that is)]  [Spotted Dick]  [Starburst]  \
0                                            NaN             NaN          NaN

   [Swedish Fish]  [Sweetums]  [Those odd marshmallow circus peanut things]  \
0             NaN         NaN                                           NaN

   [Three Musketeers]  [Peterson Brand Sidewalk Chalk]  [Peanut Butter Bars]  \
0                 NaN                              NaN                   NaN

   [Peanut Butter Jars]  [Trail Mix]  [Twix]  [Vicodin]  [White Bread]  \
0                   NaN          NaN     NaN        NaN            NaN

   [Whole Wheat anything]  [York Peppermint Patties]  \
0                     NaN                        NaN

  Please leave any remarks or comments regarding your choices.  \
0                                                NaN

  Please list any items not included above that give you JOY.  \
0                                                NaN

  Please list any items not included above that give you DESPAIR.  \
0                                                NaN

  Guess the number of mints in my hand. Betty or Veronica?  \
0                                     2           Veronica
```

```
  Check all that apply: "I cried tears of sadness at the end of  ____________"
\
0                         The last Republican Debate

  "That dress* that went viral early this year - when I first saw it, it was
________"  \
0                                     White and gold

   Fill in the blank: "Taylor Swift is a force for ___________"  \
0                                                NaN

  What is your favourite font?  \
0                     fuck you

  If you squint really hard, the words "Intelligent Design" would look like.  \
0                                        "Bullsh*t!"

  Fill in the blank: "Imitation is a form of ____________"  \
0                          Lazily flattering someone

  Please estimate the degree(s) of separation you have from the following
celebrities [JK Rowling]  \
0                                        3 or higher

  Please estimate the degree(s) of separation you have from the following
celebrities [JJ Abrams]  \
0                                        3 or higher

  Please estimate the degree(s) of separation you have from the following
celebrities [Beyoncé]  \
0                                        3 or higher

  Please estimate the degree(s) of separation you have from the following
celebrities [Bieber]  \
0                                        3 or higher

  Please estimate the degree(s) of separation you have from the following
celebrities [Kevin Bacon]  \
0                                        3 or higher

  Please estimate the degree(s) of separation you have from the following
celebrities [Francis Bacon (1561 - 1626)]  \
0                                                  1

   [Sea-salt flavored stuff, probably chocolate, since this is the "it" flavor
of the year]  \
0                                                 NaN

   [Necco Wafers] Which day do you prefer, Friday or Sunday?  \
0             NaN                                       NaN

   Please estimate the degrees of separation you have from the following folks
[Bruce Lee]  \
0                                                 NaN

   Please estimate the degrees of separation you have from the following folks
[JK Rowling]  \
0                                                 NaN

   Please estimate the degrees of separation you have from the following folks
[Malala Yousafzai]  \
0                                                 NaN

   Please estimate the degrees of separation you have from the following folks
[Thom Yorke]  \
0                                                 NaN

   Please estimate the degrees of separation you have from the following folks
[JJ Abrams]  \
0                                                 NaN

   Please estimate the degrees of separation you have from the following folks
[Hillary Clinton]  \
0                                                 NaN

   Please estimate the degrees of separation you have from the following folks
[Donald Trump]  \
0                                                 NaN

   Please estimate the degrees of separation you have from the following folks
[Beyoncé Knowles]
0                                                 NaN
```

[544]:
```python
# Using existing DataFrame (df) to extract demographic info
demographics = df[['How old are you?', 'Are you going actually going trick or␣
 ↪treating yourself?']].copy()

# Making sure the index matches the candy_df rows
demographics = demographics.iloc[:candy_df.shape[0]]

# Merging demographic data with candy ratings
merged_df = pd.merge(demographics, candy_df, left_index=True, right_index=True)
```

```
# Displaying the first row of merged DataFrame
merged_df.head(1) # Printing only one row to shrink output on PDF
```

```
[544]:  How old are you? Are you going actually going trick or treating yourself?  \
0               35                                                          No

   [Butterfinger]  [100 Grand Bar]  \
0             JOY              NaN

   [Anonymous brown globs that come in black and orange wrappers]  \
0                                            DESPAIR

   [Any full-sized candy bar]  [Black Jacks]  [Bonkers]  [Bottle Caps]  \
0                         JOY            NaN        NaN            NaN

   [Box’o’ Raisins]  [Brach products (not including candy corn)]  \
0               NaN                                       DESPAIR

   [Bubble Gum]  [Cadbury Creme Eggs]  [Candy Corn]  \
0       DESPAIR               DESPAIR           NaN

   [Vials of pure high fructose corn syrup, for main-lining into your vein]  \
0                                            DESPAIR

   [Candy that is clearly just the stuff given out for free at restaurants]  \
0                                            DESPAIR

   [Cash, or other forms of legal tender]  [Chiclets]  [Caramellos]  \
0                                     JOY         NaN           NaN

   [Snickers]  [Dark Chocolate Hershey]  [Dental paraphenalia]  [Dots]  \
0         JOY                       JOY               DESPAIR     JOY

   [Fuzzy Peaches]  [Generic Brand Acetaminophen]  [Glow sticks]  \
0         DESPAIR                        DESPAIR        DESPAIR

   [Broken glow stick]  [Goo Goo Clusters]  [Good N' Plenty]  \
0             DESPAIR                 NaN          DESPAIR

   [Gum from baseball cards]  [Gummy Bears straight up]  \
0                    DESPAIR                    DESPAIR

   [Creepy Religious comics/Chick Tracts]  [Healthy Fruit]  [Heath Bar]  \
0                                 DESPAIR          DESPAIR          NaN

   [Hershey’s Kissables]  [Hershey’s Milk Chocolate]  \
0                    NaN                         JOY
```

```
   [Hugs (actual physical hugs)]  [Jolly Rancher (bad flavor)]  \
0                        DESPAIR                            NaN

   [Jolly Ranchers (good flavor)]  [Kale smoothie]  [Kinder Happy Hippo]  \
0                             NaN              NaN                   NaN

   [Kit Kat]  [Hard Candy]  [Lapel Pins]  [LemonHeads]  [Licorice]  \
0        NaN           NaN           NaN           NaN         NaN

   [Licorice (not black)]  [Lindt Truffle]  [Lollipops]  [Mars]  [Mary Janes]  \
0                     NaN              NaN          NaN     NaN           NaN

   [Maynards]  [Milk Duds]  [LaffyTaffy]  [Minibags of chips]  \
0         NaN          NaN           NaN                  NaN

   [JoyJoy (Mit Iodine)]  [Reggie Jackson Bar]  [Pixy Stix]  [Nerds]  \
0                    NaN                   NaN          NaN      NaN

   [Nestle Crunch]  [Now'n'Laters]  [Pencils]  [Milky Way]  \
0              NaN             NaN        NaN          JOY

   [Reese’s Peanut Butter Cups]  [Tolberone something or other]  [Runts]  \
0                           JOY                             NaN      NaN

   [Junior Mints]  [Senior Mints]  [Mint Kisses]  [Mint Juleps]  \
0             JOY             NaN            NaN            NaN

   [Mint Leaves]  [Peanut M&M’s]  [Regular M&Ms]  [Mint M&Ms]  [Ribbon candy]  \
0            NaN             JOY             JOY          NaN             NaN

   [Rolos]  [Skittles]  [Smarties (American)]  [Smarties (Commonwealth)]  \
0      JOY         NaN                    JOY                        NaN

   [Chick-o-Sticks (we don’t know what that is)]  [Spotted Dick]  [Starburst]  \
0                                            NaN             NaN          NaN

   [Swedish Fish]  [Sweetums]  [Those odd marshmallow circus peanut things]  \
0             NaN         NaN                                           NaN

   [Three Musketeers]  [Peterson Brand Sidewalk Chalk]  [Peanut Butter Bars]  \
0                 NaN                              NaN                   NaN

   [Peanut Butter Jars]  [Trail Mix]  [Twix]  [Vicodin]  [White Bread]  \
0                   NaN          NaN     NaN        NaN            NaN

   [Whole Wheat anything]  [York Peppermint Patties]  \
0                     NaN                        NaN

  Please leave any remarks or comments regarding your choices.  \
0                                                NaN

  Please list any items not included above that give you JOY.  \
0                                                NaN

  Please list any items not included above that give you DESPAIR.  \
0                                                NaN

  Guess the number of mints in my hand. Betty or Veronica?  \
0                                     2          Veronica

  Check all that apply: "I cried tears of sadness at the end of  ____________"
\
0                         The last Republican Debate

  "That dress* that went viral early this year - when I first saw it, it was
________"  \
0                                     White and gold

   Fill in the blank: "Taylor Swift is a force for ___________"  \
0                                                NaN

  What is your favourite font?  \
0                     fuck you

  If you squint really hard, the words "Intelligent Design" would look like.  \
0                                        "Bullsh*t!"

  Fill in the blank: "Imitation is a form of ____________"  \
0                          Lazily flattering someone

  Please estimate the degree(s) of separation you have from the following
celebrities [JK Rowling]  \
0                                        3 or higher

  Please estimate the degree(s) of separation you have from the following
celebrities [JJ Abrams]  \
0                                        3 or higher

  Please estimate the degree(s) of separation you have from the following
celebrities [Beyoncé]  \
0                                        3 or higher

  Please estimate the degree(s) of separation you have from the following
```

```
celebrities [Bieber]  \
0                                         3 or higher

  Please estimate the degree(s) of separation you have from the following
celebrities [Kevin Bacon]  \
0                                         3 or higher

  Please estimate the degree(s) of separation you have from the following
celebrities [Francis Bacon (1561 - 1626)]  \
0                                                   1

   [Sea-salt flavored stuff, probably chocolate, since this is the "it" flavor
of the year]  \
0                                                 NaN

   [Necco Wafers]
0             NaN
```

[545]:
```python
# Stripping extra spaces from the column names in the DataFrame
merged_df.columns = merged_df.columns.str.strip()

# Renaming columns to more readable names
merged_df.rename(columns={
    '[Anonymous brown globs that come in black and orange wrappers]':␣
 ↪'Anonymous Candy',
    'If you squint really hard, the words "Intelligent Design" would look like.
 ↪': 'Intelligent Design Interpretation',
    'Fill in the blank: "Imitation is a form of ____________"': 'Imitation␣
 ↪Interpretation',
    'Please estimate the degree(s) of separation you have from the following␣
 ↪celebrities [JK Rowling]': 'Separation from JK Rowling',
    'Please estimate the degree(s) of separation you have from the following␣
 ↪celebrities [JJ Abrams]': 'Separation from JJ Abrams',
}, inplace=True)

# List of columns to keep (candy-related columns and responses)
columns_to_keep = [
    'How old are you?',
    'Are you going actually going trick or treating yourself?',
    '[Butterfinger]',
    '[100 Grand Bar]',
    'Anonymous Candy',
    'Imitation Interpretation',
    'Separation from JK Rowling',
    'Separation from JJ Abrams',
]
```

```
# Filtering the DataFrame to keep only the relevant columns
cleaned_df = merged_df[columns_to_keep]

# Dropping the two columns because they don't contribute to candy type servey
cleaned_df = cleaned_df.drop(columns=['Separation from JK Rowling', 'Separation␣
 ↪from JJ Abrams'])

# Displaying the cleaned DataFrame
cleaned_df.head(1) # Printing only one row to shrink output on PDF
```

[545]:
```
  How old are you? Are you going actually going trick or treating yourself?  \
0              35                                                       No

  [Butterfinger] [100 Grand Bar] Anonymous Candy   Imitation Interpretation
0            JOY             NaN         DESPAIR  Lazily flattering someone
```

0.8 Chapter 8 – Combining Data (Method 2: Merge)

To demonstrate the second method of combining data, I extracted two demographic-related columns (e.g., age and trick-or-treating status) from the original dataset and merged them with the main candy ratings DataFrame using pd.merge(). This approach simulates a real-world scenario where user data from one source (demographic information) is merged with behavioral data (candy ratings) from another, based on a shared index.

The merge was performed using an inner join (or specify the merge method used, e.g., how='inner'), which resulted in a unified DataFrame containing both the candy ratings and the demographic information.

The operation was successful and produced a cleaner dataset with consolidated information, ready for further analysis.

0.9 Chapter 10 - Grouping with Functions (Method 1: Grouping by age)

[546]:
```
# Defining age grouping function
def age_group(age):
    try:
        age = int(age)
        if age < 20:
            return "Teen"
        elif age < 30:
            return "20s"
        elif age < 40:
            return "30s"
        elif age < 50:
            return "40s"
        else:
            return "50+"
    except:
        return "Unknown"

# Applying grouping
df['age_group'] = df['How old are you?'].apply(age_group)

# Grouping and get proportion summary for a specific candy, formatted as % for␣
 ↪better readability
butterfinger_grouped = df.groupby('age_group')[' [Butterfinger]'].
 ↪value_counts(normalize=True, dropna=False)
butterfinger_formatted = butterfinger_grouped.mul(100).round(2).astype(str) +␣
 ↪'%'

# Displaying the result
print(butterfinger_formatted)
```

```
age_group   [Butterfinger]
20s         JOY                72.45%
            DESPAIR            20.75%
            NaN                  6.8%
30s         JOY                 75.9%
            DESPAIR            17.79%
            NaN                  6.3%
40s         JOY                73.81%
            DESPAIR            19.28%
            NaN                 6.92%
50+         JOY                71.95%
            DESPAIR            20.05%
            NaN                 7.99%
Teen        JOY                61.95%
            DESPAIR            31.42%
            NaN                 6.64%
Unknown     JOY                70.07%
            DESPAIR            23.24%
            NaN                 6.69%
Name: proportion, dtype: object
```

0.10 Chapter 10 - Grouping with Functions (Method 1: Grouping by Age)

To categorize respondents, I used the "Grouping with Functions" method to group individuals by age. I defined a function, age_group(), which groups people into different categories such as "Teen," "20s," "30s," "40s," "50+," and "Unknown." Additionally, each age group was divided into three response categories: JOY, DESPAIR, and NaN, with the percentage for each response calculated.

I applied this grouping function to create a new column named age_group. This allowed me to categorize each respondent and their respective response. I chose not to remove the NaN values in order to preserve the data and understand the proportion of missing responses. In doing so, I was able to track the percentage of missing responses for each group, with the highest missing value being 7.99% for the 50+ age group. Including the missing values provides a more complete view of the dataset and helps to understand any gaps in the data.

0.11 Chapter 10 - Split/Apply/Combine (Method: 2)

```
[547]: # Function to count JOY responses for each candy column
def joy_ratio(series):
    return (series == "JOY").sum() / series.notna().sum()

# Applying joy_ratio to multiple candy columns grouped by age_group
joy_by_age = df.groupby('age_group').agg({
    ' [Butterfinger]': joy_ratio,
    ' [100 Grand Bar]': joy_ratio,
    ' [Kit Kat]': joy_ratio,
    ' [Reese's Peanut Butter Cups]': joy_ratio
})

# Formatting the result into percentages for better readability
joy_by_age_percentage = joy_by_age.mul(100).round(2).astype(str) + '%'

# Reformatting to ensure it displays correctly
joy_by_age_percentage = joy_by_age_percentage.reset_index()

# Displaying the result
print(joy_by_age_percentage)
```

```
  age_group  [Butterfinger]  [100 Grand Bar]  [Kit Kat]  \
0       20s          77.74%           62.72%     89.53%
1       30s          81.01%           67.95%     91.72%
2       40s          79.29%           77.58%     92.66%
3       50+          78.21%           74.11%     86.45%
4      Teen          66.35%           45.16%     93.06%
5   Unknown          75.09%           68.31%     88.21%

   [Reese's Peanut Butter Cups]
0                        90.55%
1                        92.76%
2                        91.99%
3                        84.97%
4                        83.41%
5                        92.78%
```

0.12 Chapter 10 - Split/Apply/Combine (Method 2)

To analyze candy preferences by age group, I used the Split/Apply/Combine method. First, I grouped the data by age group (Teen, 20s, 30s, 40s, 50+, Unknown), then calculated the proportion of "JOY" responses for various candies. The results were formatted as percentages for easy comparison. From the results, we can see that the 30s and 40s age groups showed the highest JOY responses across all candies, especially for Reese's Peanut Butter Cups.

This approach helped me better understand the relationship between age and candy preferences.

0.13 Chapter 11 - Convert between string and date time (Method 1: Converting)

[548]:
```
# Converting Timestamp column to datetime
df['Timestamp'] = pd.to_datetime(df['Timestamp'])

# Checking the updated data type
print(df['Timestamp'].dtype)
```

```
datetime64[ns]
```

0.14 Chapter 11 - Convert between String and DateTime (Method 1: Converting)

The code above converts the Timestamp column from string format to a datetime object using pd.to_datetime(), allowing for easier date-based operations and analysis.

0.15 Chapter 11 - Generate date range (Method 2: Frequencies and data offsets)

[549]:
```
# Setting Timestamp as index for resampling
df.set_index('Timestamp', inplace=True)

# Counting the number of responses per week
weekly_counts = df.resample('W').size()

# Displaying the result
print(weekly_counts)
```

```
Timestamp
2015-10-25    1632
2015-11-01    3998
Freq: W-SUN, dtype: int64
```

0.16 Chapter 11 - Generate Date Range (Method 2: Frequencies and Date Offsets)

The code above generates a date range by setting the Timestamp column as the index and using resample().size() to count how many responses were recorded per week.