HW1-Ayodeji Iwayemi

February 10, 2024

0.0.1 Name of Student: Ayodeji Iwayemi

0.0.2 Data Analysis Homework 1: Pandas and Numpy

Objective: The aim of this assignment is to demonstrate your proficiency in using Jupyter Notebook, IPython, and particularly the Pandas library for data analysis.

Create a new Jupyter Notebook. Import all necessary libraries.(10 points)

Write a brief summary of your findings. Add comments and Markdown cells in your Jupyter Notebook to explain your code and results. (10 points)

[1]:
```
# Importing pandas library and aliasing it as pd for easier reference.
import pandas as pd

# Importing numpy library and aliasing it as np for easier reference.
import numpy as np

# Importing math library for mathematical operations.
import math
```

Q1 Implement a class for n-sided polygons and a class for points in a Euclidean system, namely polygon and point respectively. For example, a 4-sided polygon can be defined by 4 points P1, P2, P3, P4, and P1-P4 are each points of the form point(X,Y), and X and Y are coordinates on the X and Y axis, respectively. The edges are listed counterclockwise starting at the lower left: P1 to P2, P2 to P3, P3 to P4, and P4 to P1. The polygon class should work for polygons of any number of edges and have a function perimeter that returns its perimeter (sum of the lengths of the edges). (20points)

[2]:
```
#The Point class having one method 'def __init__(self, x, y)'
class Point:
    def __init__(self, x, y):
        """
        Initialize a point with given x and y coordinates.

        Args:
            x (float): The x-coordinate of the point.
            y (float): The y-coordinate of the point.
        """
```

```
        self.x = x
        self.y = y
#Creating and implementing the n-sided Polygon class using two methods: 'def␣
 ↪__init__(self, *points)' and 'perimeter(self)'
class Polygon:
    def __init__(self, *points):
        """
        Initialize a polygon with given points.

        Args:
            *points: Variable-length argument list of points defining the␣
 ↪polygon.
        """
        self.points = points

    def perimeter(self):
        """
        Calculate the perimeter of the polygon.

        Returns:
            float: The perimeter of the polygon.
        """
        perimeter = 0
        num_points = len(self.points)
        for i in range(num_points):
            # Calculate distance between consecutive points using Pythagorean␣
 ↪theorem
            dx = self.points[(i + 1) % num_points].x - self.points[i].x
            dy = self.points[(i + 1) % num_points].y - self.points[i].y
            perimeter += math.sqrt(dx ** 2 + dy ** 2)
        return perimeter
```

Explanation:

- The Point class represents a point in a 2D Euclidean system with x and y coordinates.
- The Polygon class represents a polygon composed of multiple points. It has a method perimeter to calculate the perimeter of the polygon.
- In the perimeter method, it iterates over each pair of consecutive points, calculates the distance between them using the Pythagorean theorem, and adds up all the distances to get the total perimeter.

Example: The perimeter of the polygon/triangle on point(1,1), point(1,2), and point(2,2) is 3.4; The perimeter of the 4-sided polygon on point(2,1), point(2,3), point(6,3), and point(4,1) is 10.8; print out these two examples. (10points)

[3]:
```
# Example usage
if __name__ == "__main__":
    # Triangle example
```

```
    triangle = Polygon(Point(1, 1), Point(1, 2), Point(2, 2))
    print("Perimeter of the triangle:", round(triangle.perimeter(), 1))

    # 4-sided polygon example
    quad = Polygon(Point(2, 1), Point(2, 3), Point(6, 3), Point(4, 1))
    print("Perimeter of the 4-sided polygon:", round(quad.perimeter(), 1))
```

```
Perimeter of the triangle: 3.4
Perimeter of the 4-sided polygon: 10.8
```

- The example usage demonstrates how to create instances of Polygon with different sets of points and calculate their perimeters.

Q2(50 point):

- 1. Use Pandas to load both data/AIS/transit_segments.csv, and data/AIS/vessel_information.csv. Show the first 5 rows of each dataset to inspect it.(10points)

[4]:
```
# Load transit_segments.csv
# The file has been downloaded into the working folder from "data/AIS/
 ↪transit_segments.csv"
transit_segments_df = pd.read_csv("transit_segments.csv")

# Display the first 5 rows of transit_segments.csv
print("Transit Segments Dataset:")
transit_segments_df.head()
```

```
Transit Segments Dataset:
```

[4]:

	mmsi	name	transit	segment	seg_length	avg_sog	min_sog	\
0	1	Us Govt Ves	1	1	5.1	13.2	9.2	
1	1	Dredge Capt Frank	1	1	13.5	18.6	10.4	
2	1	Us Gov Vessel	1	1	4.3	16.2	10.3	
3	1	Us Gov Vessel	2	1	9.2	15.4	14.5	
4	1	Dredge Capt Frank	2	1	9.2	15.4	14.6	

	max_sog	pdgt10	st_time	end_time
0	14.5	96.5	2/10/09 16:03	2/10/09 16:27
1	20.6	100.0	4/6/09 14:31	4/6/09 15:20
2	20.5	100.0	4/6/09 14:36	4/6/09 14:55
3	16.1	100.0	4/10/09 17:58	4/10/09 18:34
4	16.2	100.0	4/10/09 17:59	4/10/09 18:35

[5]:
```
# Load vessel_information.csv
# The file has been downloaded into the working folder from ""data/AIS/
 ↪vessel_information.csv"
vessel_info_df = pd.read_csv("vessel_information.csv")
```

```
# Display the first 5 rows of vessel_information.csv
print("\nVessel Information Dataset:")
vessel_info_df.head()
```

```
Vessel Information Dataset:
```

[5]:

	mmsi	num_names	names	sov	\
0	1	8	Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…	Y	
1	9	3	000000009/Raven/Shearwater	N	
2	21	1	Us Gov Vessel	Y	
3	74	2	Mcfaul/Sarah Bell	N	
4	103	3	Ron G/Us Navy Warship 103/Us Warship 103	Y	

	flag	flag_type	num_loas	loa	\
0	Unknown	Unknown	7	42.0/48.0/57.0/90.0/138.0/154.0/156.0	
1	Unknown	Unknown	2	50.0/62.0	
2	Unknown	Unknown	1	208.0	
3	Unknown	Unknown	1	155.0	
4	Unknown	Unknown	2	26.0/155.0	

	max_loa	num_types	type
0	156.0	4	Dredging/MilOps/Reserved/Towing
1	62.0	2	Pleasure/Tug
2	208.0	1	Unknown
3	155.0	1	Unknown
4	155.0	2	Tanker/Unknown

2. For data/AIS/vessel_information.csv, keep only those rows with the type value occurring for at least 100 times in the dataset. (10points)

[6]:
```
# Count the occurrences of each type value
type_counts = vessel_info_df['type'].value_counts()

# Get types occurring at least 100 times
types_to_keep = type_counts[type_counts >= 100].index.tolist()

# Filter rows based on types occurring at least 100 times
filtered_vessel_info_df = vessel_info_df[vessel_info_df['type'].
  ↪isin(types_to_keep)]

# Display the filtered dataset
print("Filtered Vessel Information Dataset:")
filtered_vessel_info_df
```

```
Filtered Vessel Information Dataset:
```

[6]:

	mmsi	num_names	names	sov	flag	\
2	21	1	Us Gov Vessel	Y	Unknown	

```
3             74          2   Mcfaul/Sarah Bell   N               Unknown
5            310          1             Arabella   N               Bermuda
6           3011          1            Charleston  N              Anguilla
7           4731          1             000004731  N  Yemen (Republic of)
…            …           …                 …  ..                    …
10762  866946820          1  Catherine Turecamo   N               Unknown
10764  888888888          1           Earl Jones   N               Unknown
10766  919191919          1                   Oi   N               Unknown
10768  975318642          1      Island Express   N               Unknown
10770  999999999          1   Triple Attraction   N               Unknown

      flag_type  num_loas       loa  max_loa  num_types      type
2       Unknown         1     208.0    208.0          1   Unknown
3       Unknown         1     155.0    155.0          1   Unknown
5       Foreign         1      47.0     47.0          1   Unknown
6       Foreign         1     160.0    160.0          1     Other
7       Foreign         1      30.0     30.0          1   Unknown
…         …       …        …      …        …       …
10762   Unknown         2  0.0/33.0     33.0          1       Tug
10764   Unknown         1      40.0     40.0          1    Towing
10766   Unknown         1      20.0     20.0          1  Pleasure
10768   Unknown         1      20.0     20.0          1    Towing
10770   Unknown         1      30.0     30.0          1  Pleasure

[9840 rows x 11 columns]
```

- Explanation: The pd.read_csv() function had been used to load the CSV file into a DataFrame object: vessel_info_df for vessel_information.csv. The value_counts() method was used to count the occurrences of each unique value in the 'type' column. The types occurring at least 100 times were filtered out by creating a list types_to_keep containing these types. The isin() method was used to filter rows in the DataFrame where the 'type' column value is in the types_to_keep list. The filtered DataFrame was assigned to filtered_vessel_info_df and printed to display the filtered dataset.

3. Merge data/AIS/vessel_information.csv and data/AIS/transit_segments.csv on the "mmsi" column using outer join. (10points)

[7]:
```
# Merge the two datasets on the "mmsi" column using an outer join
merged_df = pd.merge(vessel_info_df, transit_segments_df, on="mmsi",␣
 ↪how="outer")

# Display the merged dataset
print("Merged Dataset:")
merged_df
```

```
Merged Dataset:
```

[7]:

	mmsi	num_names	\
0	1	8.0	
1	1	8.0	
2	1	8.0	
3	1	8.0	
4	1	8.0	
…	…	…	
262521	666909000	NaN	
262522	666909000	NaN	
262523	666909000	NaN	
262524	666909000	NaN	
262525	666909000	NaN	

	names	sov	flag	\
0	Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…	Y	Unknown	
1	Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…	Y	Unknown	
2	Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…	Y	Unknown	
3	Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…	Y	Unknown	
4	Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…	Y	Unknown	
…	…	…	…	
262521	NaN	NaN	NaN	
262522	NaN	NaN	NaN	
262523	NaN	NaN	NaN	
262524	NaN	NaN	NaN	
262525	NaN	NaN	NaN	

	flag_type	num_loas	loa	max_loa	\
0	Unknown	7.0	42.0/48.0/57.0/90.0/138.0/154.0/156.0	156.0	
1	Unknown	7.0	42.0/48.0/57.0/90.0/138.0/154.0/156.0	156.0	
2	Unknown	7.0	42.0/48.0/57.0/90.0/138.0/154.0/156.0	156.0	
3	Unknown	7.0	42.0/48.0/57.0/90.0/138.0/154.0/156.0	156.0	
4	Unknown	7.0	42.0/48.0/57.0/90.0/138.0/154.0/156.0	156.0	
…	…	…	…	…	
262521	NaN	NaN	NaN	NaN	
262522	NaN	NaN	NaN	NaN	
262523	NaN	NaN	NaN	NaN	
262524	NaN	NaN	NaN	NaN	
262525	NaN	NaN	NaN	NaN	

	num_types	…	name	transit	segment	seg_length	\
0	4.0	…	Us Govt Ves	1	1	5.1	
1	4.0	…	Dredge Capt Frank	1	1	13.5	
2	4.0	…	Us Gov Vessel	1	1	4.3	
3	4.0	…	Us Gov Vessel	2	1	9.2	
4	4.0	…	Dredge Capt Frank	2	1	9.2	
…	…	…	…	…	…	…	
262521	NaN	…	Cg213	1	1	69.7	

```
262522        NaN  …              Cg204       1        1        37.4
262523        NaN  …              Cg204       2        1        20.8
262524        NaN  …              Cg204       3        1        49.4
262525        NaN  …              Cg204       4        1        30.9

        avg_sog  min_sog  max_sog  pdgt10         st_time         end_time
0          13.2      9.2     14.5    96.5   2/10/09 16:03   2/10/09 16:27
1          18.6     10.4     20.6   100.0    4/6/09 14:31    4/6/09 15:20
2          16.2     10.3     20.5   100.0    4/6/09 14:36    4/6/09 14:55
3          15.4     14.5     16.1   100.0   4/10/09 17:58   4/10/09 18:34
4          15.4     14.6     16.2   100.0   4/10/09 17:59   4/10/09 18:35
…             …        …        …       …              …               …
262521      8.9      0.1     16.9    76.4   11/3/08 12:28   11/3/08 22:02
262522      5.3      0.0     11.5    45.2   11/8/08 15:38   11/8/08 22:51
262523     10.7      0.0     15.5    76.9   11/9/08 14:14   11/9/08 16:11
262524      9.3      0.0     15.2    60.1  11/10/08 19:48   11/11/08 1:06
262525      8.7      0.1     49.1    96.3  11/11/08 16:29  11/11/08 19:52

[262526 rows x 21 columns]
```

4. If you are not allowed to call the inner join provided by Pandas but have the above outer join results, how to get the results of inner join? You can use other functions provided by Pandas (but not a function that directly implements the inner join). (10points)

[8]:
```
# The first thing is to observe each of the outer-merged tables for any null value
# Let's look through the vessel_info_df if it has any null values
vessel_info_df.isnull().any()
```

[8]:
```
mmsi         False
num_names    False
names        False
sov          False
flag         False
flag_type    False
num_loas     False
loa          False
max_loa      False
num_types    False
type         False
dtype: bool
```

[9]:
```
# Secondly, Let's look through the transit_segments_df if it has any null values

transit_segments_df.isnull().any()
```

[9]:
```
mmsi          False
name          False
transit       False
segment       False
seg_length    False
avg_sog       False
min_sog       False
max_sog       False
pdgt10        False
st_time       False
end_time      False
dtype: bool
```

[10]:
```
# Thirdly, Let's look through the transit_segments_df if it has any null values

merged_df.isnull().any()
```

[10]:
```
mmsi          False
num_names      True
names          True
sov            True
flag           True
flag_type      True
num_loas       True
loa            True
max_loa        True
num_types      True
type           True
name          False
transit       False
segment       False
seg_length    False
avg_sog       False
min_sog       False
max_sog       False
pdgt10        False
st_time       False
end_time      False
dtype: bool
```

- From the results above, neither of the two tables that was merged has a missing data whereas in the outer-merged result, ten (10) columns have NaN.
- Therefore, any null row should be dropped to achieve the result of an inner join without using the inner join function.

[11]:
```
# Note: Filtering  the outer-joined DataFrame to retain only rows with non-null␣
↪values in the 'mmsi' columns would result in the already got outer-merged␣
↪dataframe
```

```
'''
# Assuming merged_df is the outer-joined DataFrame resulting from pd.merge()

# Filter out rows with missing values in the columns used for the join
inner_joined_df = merged_df.dropna(subset=['mmsi'])

'''
# Filter the outer-joined DataFrame (named "merged_df") to retain only rows␣
 ↪with non-null values
inner_join_result = merged_df.dropna()

# Display the result of the inner join
print("Inner Join Result:")
inner_join_result
```

```
Inner Join Result:
```

[11]:

```
             mmsi  num_names  \
0               1        8.0
1               1        8.0
2               1        8.0
3               1        8.0
4               1        8.0
…             …        …
262348  999999999        1.0
262349  999999999        1.0
262350  999999999        1.0
262351  999999999        1.0
262352  999999999        1.0

                                                    names sov     flag  \
0       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
1       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
2       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
3       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
4       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
…                                                     …  ..       …
262348                                 Triple Attraction   N  Unknown
262349                                 Triple Attraction   N  Unknown
262350                                 Triple Attraction   N  Unknown
262351                                 Triple Attraction   N  Unknown
262352                                 Triple Attraction   N  Unknown

       flag_type  num_loas                                  loa  max_loa  \
0        Unknown       7.0  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
1        Unknown       7.0  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
2        Unknown       7.0  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
```

```
3         Unknown       7.0  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
4         Unknown       7.0  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
…           …       …                                       …      …
262348    Unknown       1.0                                    30.0     30.0
262349    Unknown       1.0                                    30.0     30.0
262350    Unknown       1.0                                    30.0     30.0
262351    Unknown       1.0                                    30.0     30.0
262352    Unknown       1.0                                    30.0     30.0

        num_types  …               name transit  segment  seg_length  \
0             4.0  …        Us Govt Ves       1        1         5.1
1             4.0  …  Dredge Capt Frank       1        1        13.5
2             4.0  …      Us Gov Vessel       1        1         4.3
3             4.0  …      Us Gov Vessel       2        1         9.2
4             4.0  …  Dredge Capt Frank       2        1         9.2
…           …  …                 …     …      …        …
262348        1.0  …  Triple Attraction       3        1         5.3
262349        1.0  …  Triple Attraction       4        1        18.7
262350        1.0  …  Triple Attraction       6        1        17.4
262351        1.0  …  Triple Attraction       7        1        31.5
262352        1.0  …  Triple Attraction       8        1        19.8

        avg_sog  min_sog  max_sog  pdgt10        st_time       end_time
0          13.2      9.2     14.5    96.5  2/10/09 16:03  2/10/09 16:27
1          18.6     10.4     20.6   100.0   4/6/09 14:31   4/6/09 15:20
2          16.2     10.3     20.5   100.0   4/6/09 14:36   4/6/09 14:55
3          15.4     14.5     16.1   100.0  4/10/09 17:58  4/10/09 18:34
4          15.4     14.6     16.2   100.0  4/10/09 17:59  4/10/09 18:35
…           …       …        …      …             …             …
262348     20.0     19.6     20.4   100.0  6/15/10 12:49  6/15/10 13:05
262349     19.2     18.4     19.9   100.0  6/15/10 21:32  6/15/10 22:29
262350     17.0     14.7     18.4   100.0  6/17/10 19:16  6/17/10 20:17
262351     14.2     13.4     15.1   100.0   6/18/10 2:52   6/18/10 5:03
262352     18.6     16.1     19.5   100.0  6/18/10 10:19  6/18/10 11:22

[262353 rows x 21 columns]
```

Now directly call the inner join provided by Pandas, check whether your results above are exactly the same.(10points)

[12]:
```
# Perform inner join directly using pandas merge
inner_join_result_pandas = pd.merge(vessel_info_df, transit_segments_df,␣
↪on="mmsi", how="inner")

# Display the result of the inner join
print("Inner Join Result (Direct Pandas):")
inner_join_result_pandas
```

Inner Join Result (Direct Pandas):

[12]:

```
             mmsi  num_names  \
0               1          8
1               1          8
2               1          8
3               1          8
4               1          8
…             …        …
262348  999999999          1
262349  999999999          1
262350  999999999          1
262351  999999999          1
262352  999999999          1

                                                    names sov     flag  \
0       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
1       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
2       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
3       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
4       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
…                                                     …  ..       …
262348                                Triple Attraction   N  Unknown
262349                                Triple Attraction   N  Unknown
262350                                Triple Attraction   N  Unknown
262351                                Triple Attraction   N  Unknown
262352                                Triple Attraction   N  Unknown

       flag_type  num_loas                                      loa  max_loa  \
0        Unknown         7  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
1        Unknown         7  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
2        Unknown         7  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
3        Unknown         7  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
4        Unknown         7  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
…          …       …                                         …      …
262348   Unknown         1                                  30.0     30.0
262349   Unknown         1                                  30.0     30.0
262350   Unknown         1                                  30.0     30.0
262351   Unknown         1                                  30.0     30.0
262352   Unknown         1                                  30.0     30.0

        num_types  …              name transit  segment  seg_length  \
0               4  …       Us Govt Ves       1        1         5.1
1               4  …  Dredge Capt Frank       1        1        13.5
2               4  …      Us Gov Vessel       1        1         4.3
3               4  …      Us Gov Vessel       2        1         9.2
4               4  …  Dredge Capt Frank       2        1         9.2
```

```
…            …  …                    …      …       …          …
262348           1  …  Triple Attraction       3         1          5.3
262349           1  …  Triple Attraction       4         1         18.7
262350           1  …  Triple Attraction       6         1         17.4
262351           1  …  Triple Attraction       7         1         31.5
262352           1  …  Triple Attraction       8         1         19.8

        avg_sog  min_sog  max_sog  pdgt10        st_time       end_time
0          13.2      9.2     14.5    96.5  2/10/09 16:03  2/10/09 16:27
1          18.6     10.4     20.6   100.0   4/6/09 14:31   4/6/09 15:20
2          16.2     10.3     20.5   100.0   4/6/09 14:36   4/6/09 14:55
3          15.4     14.5     16.1   100.0  4/10/09 17:58  4/10/09 18:34
4          15.4     14.6     16.2   100.0  4/10/09 17:59  4/10/09 18:35
…           …        …        …      …              …              …
262348     20.0     19.6     20.4   100.0  6/15/10 12:49  6/15/10 13:05
262349     19.2     18.4     19.9   100.0  6/15/10 21:32  6/15/10 22:29
262350     17.0     14.7     18.4   100.0  6/17/10 19:16  6/17/10 20:17
262351     14.2     13.4     15.1   100.0   6/18/10 2:52   6/18/10 5:03
262352     18.6     16.1     19.5   100.0  6/18/10 10:19  6/18/10 11:22

[262353 rows x 21 columns]
```

[13]:
```
# Check if the results from both approaches are exactly the same
results_match = inner_join_result.equals(inner_join_result_pandas)
print("Results Match:", results_match)
```

```
Results Match: False
```

Lets probe into the dataframes to check the reason why they seem not to be thesame

- 1. Let's check their shapes

[14]:
```
#Let's check their dimensions(shape)
print("The shape of merged dataset without using the inner function is",␣
 ↪inner_join_result.shape)
print("The shape of merged dataset using the inner function is",␣
 ↪inner_join_result_pandas.shape )
inner_join_result.shape == inner_join_result_pandas.shape
```

```
The shape of merged dataset without using the inner function is (262353, 21)
The shape of merged dataset using the inner function is (262353, 21)
```

[14]:
```
True
```

- 2. Lets observe the data typesof each of the given tables with that of the resulting merged output (using outer join)

[15]:
```
#Display the datatypes of the vessel dataset
vessel_info_df.dtypes
```

[15]:
```
mmsi           int64
num_names      int64
names         object
sov           object
flag          object
flag_type     object
num_loas       int64
loa           object
max_loa      float64
num_types      int64
type          object
dtype: object
```

[16]:
```
#Display the datatypes of the transit dataset
transit_segments_df.dtypes
```

[16]:
```
mmsi            int64
name           object
transit         int64
segment         int64
seg_length    float64
avg_sog       float64
min_sog       float64
max_sog       float64
pdgt10        float64
st_time        object
end_time       object
dtype: object
```

[17]:
```
#Display the datatypes of the outer-merged dataset
merged_df.dtypes
```

[17]:
```
mmsi            int64
num_names     float64
names          object
sov            object
flag           object
flag_type      object
num_loas      float64
loa            object
max_loa       float64
num_types     float64
type           object
name           object
```

```
transit         int64
segment         int64
seg_length    float64
avg_sog       float64
min_sog       float64
max_sog       float64
pdgt10        float64
st_time        object
end_time       object
dtype: object
```

FINDINGS:

It was observed from the datatypes above that 'num_names', 'num_loas', and 'num_types' were of the integer vessel information datatype in the original dataframe. However, after merging, they became floating in the outer-merged dataframe.

Reason:

It indicates that there were missing values (NaN) introduced during the outer-merge operation. When NaN values are introduced into a column that contains integer values, pandas automatically converts the column to a floating-point type to accommodate the presence of NaN, as NaN is a floating-point value.

But, in the inner-merged dataset, the original datatype of each of the columns was retained.

Convert the three floating columns in the outer-merged dataframe to integers as in the original vessel information dataframe

[18]:
```
inner_join_result = merged_df.dropna()
# Convert 'num_names' column to integer type
inner_join_result.loc[:, 'num_names'] = inner_join_result['num_names'].
 ↪astype('int64')

# Convert 'num_loas' column to integer type
inner_join_result.loc[:, 'num_loas'] = inner_join_result['num_loas'].
 ↪astype('int64')

# Convert 'num_types' column to integer type
inner_join_result.loc[:, 'num_types'] = inner_join_result['num_types'].
 ↪astype('int64')
inner_join_result
```

```
C:\Users\iwaye\AppData\Local\Temp\ipykernel_13124\954214950.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

```
  inner_join_result.loc[:, 'num_names'] =
inner_join_result['num_names'].astype('int64')
C:\Users\iwaye\AppData\Local\Temp\ipykernel_13124\954214950.py:3:
DeprecationWarning: In a future version, `df.iloc[:, i] = newvals` will attempt
to set the values inplace instead of always setting a new array. To retain the
old behavior, use either `df[df.columns[i]] = newvals` or, if columns are non-
unique, `df.isetitem(i, newvals)`
  inner_join_result.loc[:, 'num_names'] =
inner_join_result['num_names'].astype('int64')
C:\Users\iwaye\AppData\Local\Temp\ipykernel_13124\954214950.py:6:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  inner_join_result.loc[:, 'num_loas'] =
inner_join_result['num_loas'].astype('int64')
C:\Users\iwaye\AppData\Local\Temp\ipykernel_13124\954214950.py:6:
DeprecationWarning: In a future version, `df.iloc[:, i] = newvals` will attempt
to set the values inplace instead of always setting a new array. To retain the
old behavior, use either `df[df.columns[i]] = newvals` or, if columns are non-
unique, `df.isetitem(i, newvals)`
  inner_join_result.loc[:, 'num_loas'] =
inner_join_result['num_loas'].astype('int64')
C:\Users\iwaye\AppData\Local\Temp\ipykernel_13124\954214950.py:9:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  inner_join_result.loc[:, 'num_types'] =
inner_join_result['num_types'].astype('int64')
C:\Users\iwaye\AppData\Local\Temp\ipykernel_13124\954214950.py:9:
DeprecationWarning: In a future version, `df.iloc[:, i] = newvals` will attempt
to set the values inplace instead of always setting a new array. To retain the
old behavior, use either `df[df.columns[i]] = newvals` or, if columns are non-
unique, `df.isetitem(i, newvals)`
  inner_join_result.loc[:, 'num_types'] =
inner_join_result['num_types'].astype('int64')
```

[18]:

	mmsi	num_names	\
0	1	8	
1	1	8	
2	1	8	
3	1	8	

```
4               1          8
…             …        …
262348  999999999          1
262349  999999999          1
262350  999999999          1
262351  999999999          1
262352  999999999          1
```

```
                                                    names sov     flag  \
0       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
1       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
2       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
3       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
4       Bil Holman Dredge/Dredge Capt Frank/Emo/Offsho…   Y  Unknown
…                                                     …  ..       …
262348                                 Triple Attraction   N  Unknown
262349                                 Triple Attraction   N  Unknown
262350                                 Triple Attraction   N  Unknown
262351                                 Triple Attraction   N  Unknown
262352                                 Triple Attraction   N  Unknown

       flag_type  num_loas                                    loa  max_loa  \
0        Unknown         7  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
1        Unknown         7  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
2        Unknown         7  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
3        Unknown         7  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
4        Unknown         7  42.0/48.0/57.0/90.0/138.0/154.0/156.0    156.0
…          …       …                                        …      …
262348   Unknown         1                                  30.0     30.0
262349   Unknown         1                                  30.0     30.0
262350   Unknown         1                                  30.0     30.0
262351   Unknown         1                                  30.0     30.0
262352   Unknown         1                                  30.0     30.0

        num_types  …              name transit  segment  seg_length  \
0               4  …       Us Govt Ves       1        1         5.1
1               4  …  Dredge Capt Frank       1        1        13.5
2               4  …      Us Gov Vessel       1        1         4.3
3               4  …      Us Gov Vessel       2        1         9.2
4               4  …  Dredge Capt Frank       2        1         9.2
…             …  …                …     …       …          …
262348          1  …  Triple Attraction       3        1         5.3
262349          1  …  Triple Attraction       4        1        18.7
262350          1  …  Triple Attraction       6        1        17.4
262351          1  …  Triple Attraction       7        1        31.5
262352          1  …  Triple Attraction       8        1        19.8
```

```
        avg_sog  min_sog  max_sog  pdgt10        st_time       end_time
0          13.2      9.2     14.5    96.5  2/10/09 16:03  2/10/09 16:27
1          18.6     10.4     20.6   100.0   4/6/09 14:31   4/6/09 15:20
2          16.2     10.3     20.5   100.0   4/6/09 14:36   4/6/09 14:55
3          15.4     14.5     16.1   100.0  4/10/09 17:58  4/10/09 18:34
4          15.4     14.6     16.2   100.0  4/10/09 17:59  4/10/09 18:35
…           …        …        …      …             …              …
262348     20.0     19.6     20.4   100.0  6/15/10 12:49  6/15/10 13:05
262349     19.2     18.4     19.9   100.0  6/15/10 21:32  6/15/10 22:29
262350     17.0     14.7     18.4   100.0  6/17/10 19:16  6/17/10 20:17
262351     14.2     13.4     15.1   100.0   6/18/10 2:52   6/18/10 5:03
262352     18.6     16.1     19.5   100.0  6/18/10 10:19  6/18/10 11:22

[262353 rows x 21 columns]
```

[19]:
```
# Check again if the results from both approaches are exactly the same
results_match = inner_join_result.equals(inner_join_result_pandas)
print("Results Match:", results_match)
```

```
Results Match: True
```

Conclusion

The results from manipulating outer-merging without using the inner join function and the results obtained using the inner join function were not thesame initially but they became thesame after identifying the reasons and performing the following operations on the outer-merged dataframe: - 1 dropping all NaN rows because the original transit and vessel information tables - 2 converting the three floating number columns in the resulting inner joined dataframe (without using the inner join function but by operating the outer-merged dataframe) to integers as in the original vessel information dataframe

[ ]:
```
# Assuming vessel_info_df and transit_segments_df are DataFrames and "mmsi" is␣
 ↪the common column
# First, filter rows from vessel_info_df where "mmsi" is in transit_segments_df
inner_merged_df = vessel_info_df[vessel_info_df["mmsi"].
 ↪isin(transit_segments_df["mmsi"])]

# Then, merge the filtered DataFrame with transit_segments_df on "mmsi"
inner_merged_df = inner_merged_df.merge(transit_segments_df, on="mmsi")

# inner_merged_df now contains only the rows where "mmsi" is present in both␣
 ↪DataFrames
```