```
In [2]: import pandas as pd
        from pandas_profiling import ProfileReport
```

```
In [4]: Complete_HWC_Data = pd.read_csv("/Users/nerdbear/Downloads/Complete_HWC_Data.csv", index
```

```
In [5]: Complete_HWC_Data["Sum of Number of Animals"] = Complete_HWC_Data["Sum of Number of Anim

        Complete_HWC_Data["Total Staff Hours"] = Complete_HWC_Data["Total Staff Hours"].astype("

        Complete_HWC_Data["Total Staff Involved"] = Complete_HWC_Data["Total Staff Involved"].as

        Complete_HWC_Data["Latitude Public"] = Complete_HWC_Data["Latitude Public"].astype("floa

        Complete_HWC_Data["Longitude Public"] = Complete_HWC_Data["Longitude Public"].astype("fl

        Complete_HWC_Data[Complete_HWC_Data.columns[20:170]] = Complete_HWC_Data[Complete_HWC_Da
```

```
In [6]: Complete_HWC_Data.head()
```

Out[6]:

	UniqueID	Incident Number	Incident Date	Field Unit	Protected Heritage Area	Incident Type	Latitude Public	Longitude Public	Wi F
0	BAN2010-0003.3	BAN2010-0003	2010-01-01	Banff Field Unit	Banff National Park of Canada	Human Wildlife Interaction	51.161093	-115.593386	
1	BAN2010-0003.2	BAN2010-0003	2010-01-01	Banff Field Unit	Banff National Park of Canada	Human Wildlife Interaction	51.161093	-115.593386	
2	BAN2010-0003.1	BAN2010-0003	2010-01-01	Banff Field Unit	Banff National Park of Canada	Human Wildlife Interaction	51.161093	-115.593386	
3	JNP2010-0011.1	JNP2010-0011	2010-01-01	Jasper Field Unit	Jasper National Park of Canada	Rescued/Recovered/Found Wildlife	53.139120	-117.964219	
4	JNP2010-0015.1	JNP2010-0015	2010-01-01	Jasper Field Unit	Jasper National Park of Canada	Attractant	53.050492	-118.073612	

5 rows × 158 columns

```
In [8]: profile = ProfileReport(Complete_HWC_Data, title="Human Wildlife Coexistence Data - EDA
```

```
In [9]: profile.to_file("HWC_Data_EDA_Report.html")
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
```

```
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
```

```
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
```

```
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
```

```
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
```

```
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
```

```
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
/Users/nerdbear/opt/anaconda3/lib/python3.9/site-packages/multimethod/__init__.py:315: P
erformanceWarning: DataFrame is highly fragmented.  This is usually the result of callin
g `frame.insert` many times, which has poor performance.  Consider joining all columns a
t once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = fr
ame.copy()`
  return func(*args, **kwargs)
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
Export report to file:   0%|          | 0/1 [00:00<?, ?it/s]
```