*# This Python 3 environment comes with many helpful analytics libraries installed*

*# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python*

*# For example, here's several helpful packages to load*

import numpy as np *# linear algebra*

import pandas as pd *# data processing, CSV file I/O (e.g. pd.read\_csv)*

import matplotlib.pyplot as plt

import seaborn as sns

*# Input data files are available in the read-only "../input/" directory*

*# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory*

import os

for dirname, \_, filenames **in** os.walk('/kaggle/input'):

for filename **in** filenames:

print(os.path.join(dirname, filename))

*# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save & Run All"*

*# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session*

/kaggle/input/healthcare-analytics/test.csv

/kaggle/input/healthcare-analytics/sample\_submmission.csv

/kaggle/input/healthcare-analytics/Train/Patient\_Profile.csv

/kaggle/input/healthcare-analytics/Train/Health\_Camp\_Detail.csv

/kaggle/input/healthcare-analytics/Train/Second\_Health\_Camp\_Attended.csv

/kaggle/input/healthcare-analytics/Train/Third\_Health\_Camp\_Attended.csv

/kaggle/input/healthcare-analytics/Train/Data\_Dictionary.xlsx

/kaggle/input/healthcare-analytics/Train/First\_Health\_Camp\_Attended.csv

/kaggle/input/healthcare-analytics/Train/Train.csv

/kaggle/input/healthcare-analytics/Train/test.csv

In [2]:

pip install xlrd

Collecting xlrd

Downloading xlrd-2.0.1-py2.py3-none-any.whl (96 kB)

|████████████████████████████████| 96 kB 2.5 MB/s

Installing collected packages: xlrd

Successfully installed xlrd-2.0.1

WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv

Note: you may need to restart the kernel to use updated packages.

In [3]:

pip install openpyxl

Collecting openpyxl

Downloading openpyxl-3.0.9-py2.py3-none-any.whl (242 kB)

|████████████████████████████████| 242 kB 5.1 MB/s

Collecting et-xmlfile

Downloading et\_xmlfile-1.1.0-py3-none-any.whl (4.7 kB)

Installing collected packages: et-xmlfile, openpyxl

Successfully installed et-xmlfile-1.1.0 openpyxl-3.0.9

WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv

Note: you may need to restart the kernel to use updated packages.

1. Data preparation and cleaning

a.1 Loading the Pandas files.

In [4]:

data\_dictionary = pd.read\_excel("../input/healthcare-analytics/Train/Data\_Dictionary.xlsx")

first\_health\_camp = pd.read\_csv("../input/healthcare-analytics/Train/First\_Health\_Camp\_Attended.csv")

health\_camp\_details = pd.read\_csv("../input/healthcare-analytics/Train/Health\_Camp\_Detail.csv")

patient\_profile = pd.read\_csv("../input/healthcare-analytics/Train/Patient\_Profile.csv")

second\_health\_camp = pd.read\_csv("../input/healthcare-analytics/Train/Second\_Health\_Camp\_Attended.csv")

third\_health\_camp = pd.read\_csv("../input/healthcare-analytics/Train/Third\_Health\_Camp\_Attended.csv")

train = pd.read\_csv("../input/healthcare-analytics/Train/Train.csv")

test = pd.read\_csv("../input/healthcare-analytics/Train/test.csv")

a.2 Reading all the files

To make relevant joints

In [5]:

data\_dictionary.head()

Out[5]:

|  | Details of the Files |
| --- | --- |
| 0 | Health\_Camp\_Detail.csv – File containing Healt... |
| 1 | Train.csv – File containing registration detai... |
| 2 | Patient\_Profile.csv – This file contains Patie... |
| 3 | First\_Health\_Camp\_Attended.csv – This file con... |
| 4 | Second\_Health\_Camp\_Attended.csv - This file co... |

In [6]:

first\_health\_camp.head()

Out[6]:

|  | Patient\_ID | Health\_Camp\_ID | Donation | Health\_Score | Unnamed: 4 |
| --- | --- | --- | --- | --- | --- |
| 0 | 506181 | 6560 | 40 | 0.439024 | NaN |
| 1 | 494977 | 6560 | 20 | 0.097561 | NaN |
| 2 | 518680 | 6560 | 10 | 0.048780 | NaN |
| 3 | 509916 | 6560 | 30 | 0.634146 | NaN |
| 4 | 488006 | 6560 | 20 | 0.024390 | NaN |

In [7]:

health\_camp\_details.head()

Out[7]:

|  | Health\_Camp\_ID | Camp\_Start\_Date | Camp\_End\_Date | Category1 | Category2 | Category3 |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 6560 | 16-Aug-03 | 20-Aug-03 | First | B | 2 |
| 1 | 6530 | 16-Aug-03 | 28-Oct-03 | First | C | 2 |
| 2 | 6544 | 03-Nov-03 | 15-Nov-03 | First | F | 1 |
| 3 | 6585 | 22-Nov-03 | 05-Dec-03 | First | E | 2 |
| 4 | 6561 | 30-Nov-03 | 18-Dec-03 | First | E | 1 |

In [8]:

patient\_profile.head()

Out[8]:

|  | Patient\_ID | Online\_Follower | LinkedIn\_Shared | Twitter\_Shared | Facebook\_Shared | Income | Education\_Score | Age | First\_Interaction | City\_Type | Employer\_Category |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 516956 | 0 | 0 | 0 | 0 | 1 | 90 | 39 | 18-Jun-03 | NaN | Software Industry |
| 1 | 507733 | 0 | 0 | 0 | 0 | 1 | None | 40 | 20-Jul-03 | H | Software Industry |
| 2 | 508307 | 0 | 0 | 0 | 0 | 3 | 87 | 46 | 02-Nov-02 | D | BFSI |
| 3 | 512612 | 0 | 0 | 0 | 0 | 1 | 75 | 47 | 02-Nov-02 | D | Education |
| 4 | 521075 | 0 | 0 | 0 | 0 | 3 | None | 80 | 24-Nov-02 | H | Others |

In [9]:

second\_health\_camp.head()

Out[9]:

|  | Patient\_ID | Health\_Camp\_ID | Health Score |
| --- | --- | --- | --- |
| 0 | 526631 | 6536 | 0.875136 |
| 1 | 509122 | 6536 | 0.755700 |
| 2 | 498864 | 6536 | 0.673181 |
| 3 | 515398 | 6536 | 0.722041 |
| 4 | 504624 | 6536 | 0.464712 |

In [10]:

third\_health\_camp.head()

Out[10]:

|  | Patient\_ID | Health\_Camp\_ID | Number\_of\_stall\_visited | Last\_Stall\_Visited\_Number |
| --- | --- | --- | --- | --- |
| 0 | 517875 | 6527 | 3 | 1 |
| 1 | 504692 | 6578 | 1 | 1 |
| 2 | 504692 | 6527 | 3 | 1 |
| 3 | 493167 | 6527 | 4 | 4 |
| 4 | 510954 | 6528 | 2 | 2 |

In [11]:

train.head()

Out[11]:

|  | Patient\_ID | Health\_Camp\_ID | Registration\_Date | Var1 | Var2 | Var3 | Var4 | Var5 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 489652 | 6578 | 10-Sep-05 | 4 | 0 | 0 | 0 | 2 |
| 1 | 507246 | 6578 | 18-Aug-05 | 45 | 5 | 0 | 0 | 7 |
| 2 | 523729 | 6534 | 29-Apr-06 | 0 | 0 | 0 | 0 | 0 |
| 3 | 524931 | 6535 | 07-Feb-04 | 0 | 0 | 0 | 0 | 0 |
| 4 | 521364 | 6529 | 28-Feb-06 | 15 | 1 | 0 | 0 | 7 |

In [12]:

patient\_profile.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 37633 entries, 0 to 37632

Data columns (total 11 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Patient\_ID 37633 non-null int64

1 Online\_Follower 37633 non-null int64

2 LinkedIn\_Shared 37633 non-null int64

3 Twitter\_Shared 37633 non-null int64

4 Facebook\_Shared 37633 non-null int64

5 Income 37633 non-null object

6 Education\_Score 37633 non-null object

7 Age 37633 non-null object

8 First\_Interaction 37633 non-null object

9 City\_Type 14249 non-null object

10 Employer\_Category 2840 non-null object

dtypes: int64(5), object(6)

memory usage: 3.2+ MB

In [13]:

patient\_profile.describe()

Out[13]:

|  | Patient\_ID | Online\_Follower | LinkedIn\_Shared | Twitter\_Shared | Facebook\_Shared |
| --- | --- | --- | --- | --- | --- |
| count | 37633.000000 | 37633.000000 | 37633.000000 | 37633.000000 | 37633.000000 |
| mean | 507148.408338 | 0.022533 | 0.027077 | 0.021603 | 0.023543 |
| std | 12411.747993 | 0.148412 | 0.162311 | 0.145387 | 0.151623 |
| min | 485678.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 25% | 496393.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 50% | 507104.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 75% | 517882.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| max | 528657.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 |

Calculating the number of numeric columns in patient profile dataset

In [14]:

numerics = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']

numerical\_col\_patient\_profile = patient\_profile.select\_dtypes(include=numerics)

numerical\_col\_patient\_profile

Out[14]:

|  | Patient\_ID | Online\_Follower | LinkedIn\_Shared | Twitter\_Shared | Facebook\_Shared |
| --- | --- | --- | --- | --- | --- |
| 0 | 516956 | 0 | 0 | 0 | 0 |
| 1 | 507733 | 0 | 0 | 0 | 0 |
| 2 | 508307 | 0 | 0 | 0 | 0 |
| 3 | 512612 | 0 | 0 | 0 | 0 |
| 4 | 521075 | 0 | 0 | 0 | 0 |
| ... | ... | ... | ... | ... | ... |
| 37628 | 518600 | 0 | 0 | 0 | 0 |
| 37629 | 509515 | 0 | 0 | 0 | 0 |
| 37630 | 510713 | 0 | 0 | 0 | 0 |
| 37631 | 493667 | 0 | 0 | 0 | 0 |
| 37632 | 498051 | 0 | 0 | 0 | 0 |

37633 rows × 5 columns

In [15]:

numerical\_col\_patient\_profile.columns

Out[15]:

Index(['Patient\_ID', 'Online\_Follower', 'LinkedIn\_Shared', 'Twitter\_Shared',

'Facebook\_Shared'],

dtype='object')

In [16]:

patient\_profile.isnull().sum() *#checking missing values in patient profile*

Out[16]:

Patient\_ID 0

Online\_Follower 0

LinkedIn\_Shared 0

Twitter\_Shared 0

Facebook\_Shared 0

Income 0

Education\_Score 0

Age 0

First\_Interaction 0

City\_Type 23384

Employer\_Category 34793

dtype: int64

a.3 Merging these different dataframes into single dataframe for further analysis

In [17]:

merged\_details = pd.merge(right = patient\_profile, left = first\_health\_camp, on="Patient\_ID")

merged\_details.info()

<class 'pandas.core.frame.DataFrame'>

Int64Index: 6218 entries, 0 to 6217

Data columns (total 15 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Patient\_ID 6218 non-null int64

1 Health\_Camp\_ID 6218 non-null int64

2 Donation 6218 non-null int64

3 Health\_Score 6218 non-null float64

4 Unnamed: 4 0 non-null float64

5 Online\_Follower 6218 non-null int64

6 LinkedIn\_Shared 6218 non-null int64

7 Twitter\_Shared 6218 non-null int64

8 Facebook\_Shared 6218 non-null int64

9 Income 6218 non-null object

10 Education\_Score 6218 non-null object

11 Age 6218 non-null object

12 First\_Interaction 6218 non-null object

13 City\_Type 4451 non-null object

14 Employer\_Category 2300 non-null object

dtypes: float64(2), int64(7), object(6)

memory usage: 777.2+ KB

In [18]:

merged\_details = merged\_details.merge(health\_camp\_details, on = "Health\_Camp\_ID" )

merged\_details.info()

<class 'pandas.core.frame.DataFrame'>

Int64Index: 6218 entries, 0 to 6217

Data columns (total 20 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Patient\_ID 6218 non-null int64

1 Health\_Camp\_ID 6218 non-null int64

2 Donation 6218 non-null int64

3 Health\_Score 6218 non-null float64

4 Unnamed: 4 0 non-null float64

5 Online\_Follower 6218 non-null int64

6 LinkedIn\_Shared 6218 non-null int64

7 Twitter\_Shared 6218 non-null int64

8 Facebook\_Shared 6218 non-null int64

9 Income 6218 non-null object

10 Education\_Score 6218 non-null object

11 Age 6218 non-null object

12 First\_Interaction 6218 non-null object

13 City\_Type 4451 non-null object

14 Employer\_Category 2300 non-null object

15 Camp\_Start\_Date 6218 non-null object

16 Camp\_End\_Date 6218 non-null object

17 Category1 6218 non-null object

18 Category2 6218 non-null object

19 Category3 6218 non-null int64

dtypes: float64(2), int64(8), object(10)

memory usage: 1020.1+ KB

In [19]:

merged\_details = merged\_details.merge(second\_health\_camp, on="Patient\_ID")

merged\_details.info()

<class 'pandas.core.frame.DataFrame'>

Int64Index: 5212 entries, 0 to 5211

Data columns (total 22 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Patient\_ID 5212 non-null int64

1 Health\_Camp\_ID\_x 5212 non-null int64

2 Donation 5212 non-null int64

3 Health\_Score 5212 non-null float64

4 Unnamed: 4 0 non-null float64

5 Online\_Follower 5212 non-null int64

6 LinkedIn\_Shared 5212 non-null int64

7 Twitter\_Shared 5212 non-null int64

8 Facebook\_Shared 5212 non-null int64

9 Income 5212 non-null object

10 Education\_Score 5212 non-null object

11 Age 5212 non-null object

12 First\_Interaction 5212 non-null object

13 City\_Type 4470 non-null object

14 Employer\_Category 2948 non-null object

15 Camp\_Start\_Date 5212 non-null object

16 Camp\_End\_Date 5212 non-null object

17 Category1 5212 non-null object

18 Category2 5212 non-null object

19 Category3 5212 non-null int64

20 Health\_Camp\_ID\_y 5212 non-null int64

21 Health Score 5212 non-null float64

dtypes: float64(3), int64(9), object(10)

memory usage: 936.5+ KB

In [20]:

merged\_details = merged\_details.merge( third\_health\_camp ,on = "Patient\_ID")

b.1 Looking at all the information present in the merged dataframe.

In [21]:

merged\_details.head()

Out[21]:

|  | Patient\_ID | Health\_Camp\_ID\_x | Donation | Health\_Score | Unnamed: 4 | Online\_Follower | LinkedIn\_Shared | Twitter\_Shared | Facebook\_Shared | Income | ... | Camp\_Start\_Date | Camp\_End\_Date | Category1 | Category2 | Category3 | Health\_Camp\_ID\_y | Health Score | Health\_Camp\_ID | Number\_of\_stall\_visited | Last\_Stall\_Visited\_Number |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 494977 | 6560 | 20 | 0.097561 | NaN | 1 | 1 | 1 | 0 | 2 | ... | 16-Aug-03 | 20-Aug-03 | First | B | 2 | 6536 | 0.673181 | 6578 | 5 | 3 |
| 1 | 494977 | 6560 | 20 | 0.097561 | NaN | 1 | 1 | 1 | 0 | 2 | ... | 16-Aug-03 | 20-Aug-03 | First | B | 2 | 6536 | 0.673181 | 6527 | 2 | 1 |
| 2 | 494977 | 6560 | 20 | 0.097561 | NaN | 1 | 1 | 1 | 0 | 2 | ... | 16-Aug-03 | 20-Aug-03 | First | B | 2 | 6555 | 0.615827 | 6578 | 5 | 3 |
| 3 | 494977 | 6560 | 20 | 0.097561 | NaN | 1 | 1 | 1 | 0 | 2 | ... | 16-Aug-03 | 20-Aug-03 | First | B | 2 | 6555 | 0.615827 | 6527 | 2 | 1 |
| 4 | 494977 | 6585 | 60 | 0.733990 | NaN | 1 | 1 | 1 | 0 | 2 | ... | 22-Nov-03 | 05-Dec-03 | First | E | 2 | 6536 | 0.673181 | 6578 | 5 | 3 |

5 rows × 25 columns

In [22]:

merged\_details.describe()

Out[22]:

|  | Patient\_ID | Health\_Camp\_ID\_x | Donation | Health\_Score | Unnamed: 4 | Online\_Follower | LinkedIn\_Shared | Twitter\_Shared | Facebook\_Shared | Category3 | Health\_Camp\_ID\_y | Health Score | Health\_Camp\_ID | Number\_of\_stall\_visited | Last\_Stall\_Visited\_Number |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| count | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 0.0 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 |
| mean | 507208.149023 | 6551.725378 | 32.880856 | 0.525072 | NaN | 0.175028 | 0.204168 | 0.164515 | 0.155293 | 1.991332 | 6536.001660 | 0.607178 | 6544.733678 | 3.329768 | 2.667650 |
| std | 12311.461570 | 17.691238 | 25.985824 | 0.281859 | NaN | 0.380026 | 0.403130 | 0.370776 | 0.362217 | 0.092708 | 10.987919 | 0.266534 | 22.747020 | 1.765143 | 1.580357 |
| min | 485720.000000 | 6524.000000 | 10.000000 | 0.001667 | NaN | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 1.000000 | 6523.000000 | 0.058993 | 6527.000000 | 0.000000 | 0.000000 |
| 25% | 496138.000000 | 6538.000000 | 20.000000 | 0.296296 | NaN | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 2.000000 | 6529.000000 | 0.399373 | 6527.000000 | 2.000000 | 1.000000 |
| 50% | 507904.000000 | 6543.000000 | 30.000000 | 0.549327 | NaN | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 2.000000 | 6534.000000 | 0.627865 | 6528.000000 | 3.000000 | 2.000000 |
| 75% | 518043.000000 | 6570.000000 | 40.000000 | 0.756793 | NaN | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 2.000000 | 6549.000000 | 0.845597 | 6578.000000 | 5.000000 | 4.000000 |
| max | 528589.000000 | 6586.000000 | 280.000000 | 1.000000 | NaN | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 2.000000 | 6555.000000 | 1.000000 | 6578.000000 | 7.000000 | 6.000000 |

In [23]:

merged\_details.columns

Out[23]:

Index(['Patient\_ID', 'Health\_Camp\_ID\_x', 'Donation', 'Health\_Score',

'Unnamed: 4', 'Online\_Follower', 'LinkedIn\_Shared', 'Twitter\_Shared',

'Facebook\_Shared', 'Income', 'Education\_Score', 'Age',

'First\_Interaction', 'City\_Type', 'Employer\_Category',

'Camp\_Start\_Date', 'Camp\_End\_Date', 'Category1', 'Category2',

'Category3', 'Health\_Camp\_ID\_y', 'Health Score', 'Health\_Camp\_ID',

'Number\_of\_stall\_visited', 'Last\_Stall\_Visited\_Number'],

dtype='object')

In [24]:

merged\_details.info()

<class 'pandas.core.frame.DataFrame'>

Int64Index: 5422 entries, 0 to 5421

Data columns (total 25 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Patient\_ID 5422 non-null int64

1 Health\_Camp\_ID\_x 5422 non-null int64

2 Donation 5422 non-null int64

3 Health\_Score 5422 non-null float64

4 Unnamed: 4 0 non-null float64

5 Online\_Follower 5422 non-null int64

6 LinkedIn\_Shared 5422 non-null int64

7 Twitter\_Shared 5422 non-null int64

8 Facebook\_Shared 5422 non-null int64

9 Income 5422 non-null object

10 Education\_Score 5422 non-null object

11 Age 5422 non-null object

12 First\_Interaction 5422 non-null object

13 City\_Type 4782 non-null object

14 Employer\_Category 3380 non-null object

15 Camp\_Start\_Date 5422 non-null object

16 Camp\_End\_Date 5422 non-null object

17 Category1 5422 non-null object

18 Category2 5422 non-null object

19 Category3 5422 non-null int64

20 Health\_Camp\_ID\_y 5422 non-null int64

21 Health Score 5422 non-null float64

22 Health\_Camp\_ID 5422 non-null int64

23 Number\_of\_stall\_visited 5422 non-null int64

24 Last\_Stall\_Visited\_Number 5422 non-null int64

dtypes: float64(3), int64(12), object(10)

memory usage: 1.1+ MB

b.2 Looking at all the columns with numerical values (as they will be needed further)

In [25]:

numerics = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']

numerical\_merged\_cols = merged\_details.select\_dtypes(include=numerics)

numerical\_merged\_cols

Out[25]:

|  | Patient\_ID | Health\_Camp\_ID\_x | Donation | Health\_Score | Unnamed: 4 | Online\_Follower | LinkedIn\_Shared | Twitter\_Shared | Facebook\_Shared | Category3 | Health\_Camp\_ID\_y | Health Score | Health\_Camp\_ID | Number\_of\_stall\_visited | Last\_Stall\_Visited\_Number |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 494977 | 6560 | 20 | 0.097561 | NaN | 1 | 1 | 1 | 0 | 2 | 6536 | 0.673181 | 6578 | 5 | 3 |
| 1 | 494977 | 6560 | 20 | 0.097561 | NaN | 1 | 1 | 1 | 0 | 2 | 6536 | 0.673181 | 6527 | 2 | 1 |
| 2 | 494977 | 6560 | 20 | 0.097561 | NaN | 1 | 1 | 1 | 0 | 2 | 6555 | 0.615827 | 6578 | 5 | 3 |
| 3 | 494977 | 6560 | 20 | 0.097561 | NaN | 1 | 1 | 1 | 0 | 2 | 6555 | 0.615827 | 6527 | 2 | 1 |
| 4 | 494977 | 6585 | 60 | 0.733990 | NaN | 1 | 1 | 1 | 0 | 2 | 6536 | 0.673181 | 6578 | 5 | 3 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 5417 | 499996 | 6524 | 20 | 0.537037 | NaN | 0 | 0 | 0 | 0 | 2 | 6534 | 0.586493 | 6527 | 6 | 2 |
| 5418 | 495949 | 6524 | 30 | 0.037037 | NaN | 0 | 0 | 0 | 0 | 2 | 6523 | 0.753930 | 6527 | 3 | 4 |
| 5419 | 520328 | 6524 | 20 | 0.092593 | NaN | 0 | 0 | 0 | 0 | 2 | 6549 | 0.552962 | 6527 | 6 | 4 |
| 5420 | 517824 | 6524 | 20 | 0.648148 | NaN | 0 | 0 | 0 | 0 | 2 | 6534 | 0.610190 | 6578 | 3 | 2 |
| 5421 | 517824 | 6524 | 20 | 0.648148 | NaN | 0 | 0 | 0 | 0 | 2 | 6534 | 0.610190 | 6527 | 6 | 4 |

5422 rows × 15 columns

In [26]:

numerical\_merged\_cols.describe()

Out[26]:

|  | Patient\_ID | Health\_Camp\_ID\_x | Donation | Health\_Score | Unnamed: 4 | Online\_Follower | LinkedIn\_Shared | Twitter\_Shared | Facebook\_Shared | Category3 | Health\_Camp\_ID\_y | Health Score | Health\_Camp\_ID | Number\_of\_stall\_visited | Last\_Stall\_Visited\_Number |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| count | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 0.0 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 | 5422.000000 |
| mean | 507208.149023 | 6551.725378 | 32.880856 | 0.525072 | NaN | 0.175028 | 0.204168 | 0.164515 | 0.155293 | 1.991332 | 6536.001660 | 0.607178 | 6544.733678 | 3.329768 | 2.667650 |
| std | 12311.461570 | 17.691238 | 25.985824 | 0.281859 | NaN | 0.380026 | 0.403130 | 0.370776 | 0.362217 | 0.092708 | 10.987919 | 0.266534 | 22.747020 | 1.765143 | 1.580357 |
| min | 485720.000000 | 6524.000000 | 10.000000 | 0.001667 | NaN | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 1.000000 | 6523.000000 | 0.058993 | 6527.000000 | 0.000000 | 0.000000 |
| 25% | 496138.000000 | 6538.000000 | 20.000000 | 0.296296 | NaN | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 2.000000 | 6529.000000 | 0.399373 | 6527.000000 | 2.000000 | 1.000000 |
| 50% | 507904.000000 | 6543.000000 | 30.000000 | 0.549327 | NaN | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 2.000000 | 6534.000000 | 0.627865 | 6528.000000 | 3.000000 | 2.000000 |
| 75% | 518043.000000 | 6570.000000 | 40.000000 | 0.756793 | NaN | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 2.000000 | 6549.000000 | 0.845597 | 6578.000000 | 5.000000 | 4.000000 |
| max | 528589.000000 | 6586.000000 | 280.000000 | 1.000000 | NaN | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 2.000000 | 6555.000000 | 1.000000 | 6578.000000 | 7.000000 | 6.000000 |

C.1 Finding what all values are missing

In [27]:

merged\_details.isna() *#shows no of values in each column tha are null*

Out[27]:

|  | Patient\_ID | Health\_Camp\_ID\_x | Donation | Health\_Score | Unnamed: 4 | Online\_Follower | LinkedIn\_Shared | Twitter\_Shared | Facebook\_Shared | Income | ... | Camp\_Start\_Date | Camp\_End\_Date | Category1 | Category2 | Category3 | Health\_Camp\_ID\_y | Health Score | Health\_Camp\_ID | Number\_of\_stall\_visited | Last\_Stall\_Visited\_Number |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | False | False | False | False | True | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| 1 | False | False | False | False | True | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| 2 | False | False | False | False | True | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| 3 | False | False | False | False | True | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| 4 | False | False | False | False | True | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 5417 | False | False | False | False | True | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| 5418 | False | False | False | False | True | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| 5419 | False | False | False | False | True | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| 5420 | False | False | False | False | True | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| 5421 | False | False | False | False | True | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |

5422 rows × 25 columns

In [28]:

merged\_details.isna().sum()

Out[28]:

Patient\_ID 0

Health\_Camp\_ID\_x 0

Donation 0

Health\_Score 0

Unnamed: 4 5422

Online\_Follower 0

LinkedIn\_Shared 0

Twitter\_Shared 0

Facebook\_Shared 0

Income 0

Education\_Score 0

Age 0

First\_Interaction 0

City\_Type 640

Employer\_Category 2042

Camp\_Start\_Date 0

Camp\_End\_Date 0

Category1 0

Category2 0

Category3 0

Health\_Camp\_ID\_y 0

Health Score 0

Health\_Camp\_ID 0

Number\_of\_stall\_visited 0

Last\_Stall\_Visited\_Number 0

dtype: int64

This shows how many values in each column are missing.

Thus we see that column Unamed:4 has all the values as Null whereas City\_type has 640 missing values and Employer\_Category has 2041 missing values.

In [29]:

*#percentage of missing values in each column*

missing\_percent = merged\_details.isna().sum().sort\_values(ascending=False)/len(merged\_details)

missing\_percent\*100 *#%*

Out[29]:

Unnamed: 4 100.000000

Employer\_Category 37.661380

City\_Type 11.803762

Patient\_ID 0.000000

Number\_of\_stall\_visited 0.000000

Health\_Camp\_ID 0.000000

Health Score 0.000000

Health\_Camp\_ID\_y 0.000000

Category3 0.000000

Category2 0.000000

Category1 0.000000

Camp\_End\_Date 0.000000

Camp\_Start\_Date 0.000000

First\_Interaction 0.000000

Health\_Camp\_ID\_x 0.000000

Age 0.000000

Education\_Score 0.000000

Income 0.000000

Facebook\_Shared 0.000000

Twitter\_Shared 0.000000

LinkedIn\_Shared 0.000000

Online\_Follower 0.000000

Health\_Score 0.000000

Donation 0.000000

Last\_Stall\_Visited\_Number 0.000000

dtype: float64

Plotting a visual of all missing values.

In [30]:

plt.figure(figsize=(5.5,1))

missing\_percent[missing\_percent!=0].plot(kind='barh')

Out[30]:

<AxesSubplot:>

C.2 Since Unamed:4 is missing all the time, we will drop this column. While others are present in atleast 60% rows so retaining them.

In [31]:

merged\_details2 = merged\_details.drop('Unnamed: 4', axis=1, inplace=True)

c.3 Handling the missing data

In [32]:

merged\_details.columns

Out[32]:

Index(['Patient\_ID', 'Health\_Camp\_ID\_x', 'Donation', 'Health\_Score',

'Online\_Follower', 'LinkedIn\_Shared', 'Twitter\_Shared',

'Facebook\_Shared', 'Income', 'Education\_Score', 'Age',

'First\_Interaction', 'City\_Type', 'Employer\_Category',

'Camp\_Start\_Date', 'Camp\_End\_Date', 'Category1', 'Category2',

'Category3', 'Health\_Camp\_ID\_y', 'Health Score', 'Health\_Camp\_ID',

'Number\_of\_stall\_visited', 'Last\_Stall\_Visited\_Number'],

dtype='object')

In [33]:

*#we're choosing the important columns that may give us meaningful results*

imp\_col = ['Patient\_ID', 'Health\_Camp\_ID', 'Donation', 'Health\_Score', 'Income', 'Camp\_Start\_Date', 'Camp\_End\_Date', 'Health Score', 'Number\_of\_stall\_visited']

merged\_details[imp\_col]

Out[33]:

|  | Patient\_ID | Health\_Camp\_ID | Donation | Health\_Score | Income | Camp\_Start\_Date | Camp\_End\_Date | Health Score | Number\_of\_stall\_visited |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 494977 | 6578 | 20 | 0.097561 | 2 | 16-Aug-03 | 20-Aug-03 | 0.673181 | 5 |
| 1 | 494977 | 6527 | 20 | 0.097561 | 2 | 16-Aug-03 | 20-Aug-03 | 0.673181 | 2 |
| 2 | 494977 | 6578 | 20 | 0.097561 | 2 | 16-Aug-03 | 20-Aug-03 | 0.615827 | 5 |
| 3 | 494977 | 6527 | 20 | 0.097561 | 2 | 16-Aug-03 | 20-Aug-03 | 0.615827 | 2 |
| 4 | 494977 | 6578 | 60 | 0.733990 | 2 | 22-Nov-03 | 05-Dec-03 | 0.673181 | 5 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 5417 | 499996 | 6527 | 20 | 0.537037 | None | 13-May-05 | 21-May-05 | 0.586493 | 6 |
| 5418 | 495949 | 6527 | 30 | 0.037037 | None | 13-May-05 | 21-May-05 | 0.753930 | 3 |
| 5419 | 520328 | 6527 | 20 | 0.092593 | None | 13-May-05 | 21-May-05 | 0.552962 | 6 |
| 5420 | 517824 | 6578 | 20 | 0.648148 | None | 13-May-05 | 21-May-05 | 0.610190 | 3 |
| 5421 | 517824 | 6527 | 20 | 0.648148 | None | 13-May-05 | 21-May-05 | 0.610190 | 6 |

5422 rows × 9 columns

In the code cell above, I have created a list of columns that are crucial for subsequent steps

Since health score has two different columns, will add them.

In [34]:

merged\_details['health'] = merged\_details['Health Score'] + merged\_details['Health\_Score']

merged\_details['health']

Out[34]:

0 0.770742

1 0.770742

2 0.713388

3 0.713388

4 1.407171

...

5417 1.123530

5418 0.790967

5419 0.645555

5420 1.258338

5421 1.258338

Name: health, Length: 5422, dtype: float64

In [35]:

imp\_cols\_pred = ['Donation', 'health', 'Income', 'Number\_of\_stall\_visited', 'Employer\_Category', 'City\_Type']

Creating a separate dataframe of the columns selected in the imp\_cols\_pred list.

In [36]:

useful\_details = merged\_details[imp\_cols\_pred]

useful\_details.head()

Out[36]:

|  | Donation | health | Income | Number\_of\_stall\_visited | Employer\_Category | City\_Type |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 20 | 0.770742 | 2 | 5 | Transport | H |
| 1 | 20 | 0.770742 | 2 | 2 | Transport | H |
| 2 | 20 | 0.713388 | 2 | 5 | Transport | H |
| 3 | 20 | 0.713388 | 2 | 2 | Transport | H |
| 4 | 60 | 1.407171 | 2 | 5 | Transport | H |

In [37]:

useful\_details.info()

<class 'pandas.core.frame.DataFrame'>

Int64Index: 5422 entries, 0 to 5421

Data columns (total 6 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Donation 5422 non-null int64

1 health 5422 non-null float64

2 Income 5422 non-null object

3 Number\_of\_stall\_visited 5422 non-null int64

4 Employer\_Category 3380 non-null object

5 City\_Type 4782 non-null object

dtypes: float64(1), int64(2), object(3)

memory usage: 296.5+ KB

So employer\_count has 3380 (out of 5422) non null values And city\_type has 4782(out of 5422) non null values

In [38]:

useful\_details['Employer\_Category'].mode()

Out[38]:

0 Technology

dtype: object

In [39]:

useful\_details['City\_Type'].mode()

Out[39]:

0 H

dtype: object

So the most frequent values in Employer\_Category and City\_Type are 'Technology' and 'H' respectively. So we will fill the null places with it.

In [40]:

useful\_details.City\_Type.fillna('H', inplace=True)

/opt/conda/lib/python3.7/site-packages/pandas/core/generic.py:6392: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame

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

return self.\_update\_inplace(result)

In [41]:

useful\_details.Employer\_Category.fillna('Technology', inplace = True)

Now we will check some key parameters like income, donation, no of stalls visited and health

In [42]:

useful\_details.head()

Out[42]:

|  | Donation | health | Income | Number\_of\_stall\_visited | Employer\_Category | City\_Type |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 20 | 0.770742 | 2 | 5 | Transport | H |
| 1 | 20 | 0.770742 | 2 | 2 | Transport | H |
| 2 | 20 | 0.713388 | 2 | 5 | Transport | H |
| 3 | 20 | 0.713388 | 2 | 2 | Transport | H |
| 4 | 60 | 1.407171 | 2 | 5 | Transport | H |

In [43]:

useful\_details.info()

<class 'pandas.core.frame.DataFrame'>

Int64Index: 5422 entries, 0 to 5421

Data columns (total 6 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Donation 5422 non-null int64

1 health 5422 non-null float64

2 Income 5422 non-null object

3 Number\_of\_stall\_visited 5422 non-null int64

4 Employer\_Category 5422 non-null object

5 City\_Type 5422 non-null object

dtypes: float64(1), int64(2), object(3)

memory usage: 296.5+ KB

In [44]:

useful\_details.Donation.describe()

Out[44]:

count 5422.000000

mean 32.880856

std 25.985824

min 10.000000

25% 20.000000

50% 30.000000

75% 40.000000

max 280.000000

Name: Donation, dtype: float64

In [45]:

useful\_details.health.describe()

Out[45]:

count 5422.000000

mean 1.132249

std 0.407618

min 0.077326

25% 0.844802

50% 1.147704

75% 1.432447

max 1.985676

Name: health, dtype: float64

In [46]:

useful\_details.Income.describe()

Out[46]:

count 5422

unique 8

top 1

freq 1298

Name: Income, dtype: object

In [47]:

useful\_details.Number\_of\_stall\_visited.describe()

Out[47]:

count 5422.000000

mean 3.329768

std 1.765143

min 0.000000

25% 2.000000

50% 3.000000

75% 5.000000

max 7.000000

Name: Number\_of\_stall\_visited, dtype: float64

In [48]:

useful\_details.City\_Type.describe()

Out[48]:

count 5422

unique 9

top H

freq 1402

Name: City\_Type, dtype: object

2 Visualisation

In [49]:

for col **in** useful\_details.columns:

if useful\_details[col].dtype == 'int64':

plt.pie(useful\_details[col].value\_counts(), labels=useful\_details[col].unique())

plt.title('Piechart for **{}**'.format(col))

plt.show()

else:

plt.hist(useful\_details[col].value\_counts())

plt.title('Histogram for **{}**'.format(col))

plt.show();

In [50]:

plt.scatter(x=useful\_details['Employer\_Category'] , y=useful\_details['health'], s =100)

plt.figure(figsize=(50,30))

Out[50]:

<Figure size 3600x2160 with 0 Axes>

<Figure size 3600x2160 with 0 Axes>

In [51]:

plt.scatter(x=useful\_details['Income'], y=useful\_details['health'], s = 75)

Out[51]:

<matplotlib.collections.PathCollection at 0x7f51fa98e410>

In [52]:

sns.violinplot(y = merged\_details['health'], x = merged\_details['Camp\_Start\_Date'])

*#sns.boxplot(y = merged\_details['health'], x = merged\_details['Camp\_Start\_Date'])*

sns.set(rc={"figure.figsize":(200,3)})

In [53]:

sns.jointplot(x=merged\_details['LinkedIn\_Shared'], y=merged\_details['Twitter\_Shared'], data = merged\_details['health'], kind = 'scatter')

Out[53]:

<seaborn.axisgrid.JointGrid at 0x7f51fa7e79d0>

3 Inference

From the above visalisation of data, a striking observation I noted is that health score is not strongly dependent of the camp starting date as validated by the violinplot where barring one almost all violin plots had similar lengths and distributions.

The income-helth scatter plot shows high income patients tend to have a higher healthscore (as illustrated by the density of the scatterplot) whereas low income patients have densities distributed over a range of healthscore.

4 Moving on to ML prediction

In [54]:

merged\_dumm=pd.get\_dummies(merged\_details, prefix=None, prefix\_sep="\_",drop\_first=False)

In [55]:

x=merged\_dumm.iloc[:,:-1]

y=merged\_dumm.health

x.shape

Out[55]:

(5422, 653)

In [56]:

from sklearn.model\_selection import train\_test\_split

In [57]:

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y,test\_size = .3,random\_state =23)

from sklearn.linear\_model import LinearRegression

model=LinearRegression()

model=model.fit(x\_train,y\_train)

rsq=model.score(x\_train,y\_train)

In [58]:

rsq=model.score(x\_test,y\_test)

rsq

Out[58]:

1.0

In [ ]: