In [198…

```
'''You are employed as a data scientist by the American Census Agency.
the agency is interested in building a predictive model to estimate the death rate among cancer patients.
Your task is to clean the data in readines for the next stage of the process.

Please complete the following:

1. import the data into your jupyter notebook and analyze it in detail

2. Identify and report on the structure and any problems you find in the data

3. Identify and report missing values and duplicate samples in the data

4. Clean the data of missing values and also of duplicate values.
(Explain your choices and give justifications for any method you use to clean the data of missing values)

5. Report the structure of the final data after cleaning (number of samples and number of features)'''
```

Out[198…

```
'You are employed as a data scientist by the American Census Agency. \nthe agency is interested in building a p
redictive model to estimate the death rate among cancer patients. \nYour task is to clean the data in readines
for the next stage of the process.\n\nPlease complete the following:\n\n1. import the data into your jupyter no
tebook and analyze it in detail\n\n2. Identify and report on the structure and any problems you find in the dat
a\n\n3. Identify and report missing values and duplicate samples in the data \n\n4. Clean the data of missing v
alues and also of duplicate values. \n(Explain your choices and give justifications for any method you use to c
lean the data of missing values)\n\n5. Report the structure of the final data after cleaning (number of samples
and number of features)'
```

In [ ]:

In [334…

```
#Question 1

#import the data into your jupyter notebook and analyze it in detail

fileName = 'cancerdata.csv'
filePath = '/Users/tomisin/Dropbox/My Mac (Tomisins-MacBook-Pro.local)/Documents/MY WORKSPACE/'

import pandas as pd
fileName = 'cancerdata.csv'
df = pd.read_csv(filePath +fileName)
```

In [335…

```
pd.set_option('display.max_columns', 40)
```

```
pd.set_option('display.max_rows', 100)
```

In [336…

```
df
```

Out[336…

	Unnamed: 0	avgAnnCount	avgDeathsPerYear	TARGET_deathRate	incidenceRate	medIncome	popEst2015	povertyPercent	study
0	0	1397.000000	469.0	164.9	489.800000	61898.0	260131.0	11.2	499.
1	1	173.000000	70.0	161.3	411.600000	48127.0	NaN	18.6	23
2	2	NaN	50.0	174.7	349.700000	49348.0	21026.0	14.6	47.
3	3	427.000000	NaN	194.8	430.400000	44243.0	75882.0	NaN	342.
4	4	57.000000	NaN	144.4	350.100000	49955.0	10321.0	12.5	0.
...	...	...	...	...	...	...	...	...	
3042	3042	1962.667684	15.0	NaN	453.549422	46961.0	6343.0	12.4	0.
3043	3043	1962.667684	43.0	150.1	453.549422	48609.0	37118.0	NaN	377
3044	3044	NaN	46.0	153.9	453.549422	NaN	NaN	15.0	1968.
3045	3045	1962.667684	52.0	175.0	453.549422	50745.0	25609.0	13.3	0.
3046	3046	1962.667684	48.0	213.6	453.549422	41193.0	37030.0	13.9	0.

3047 rows × 35 columns

In [ ]:

In [337…

```
#Question 2

#Identify and report on the structure and any problems you find in the data

''' 1. The structure of the data is such that has 3074 samples and 35 features
    (two of which are object and the rest float data types)'''

''' 2. Here, judging from the difference between the mean and the 50% percentile,
    the data distribution is generally skewed except for a few features
    with near perfect normalization
    (having the so called gaussian type density curves)
    like TARGET_deathRate, medianAgemale, medianAgefemale, PctHS18_24,
    PctSomeCol18_24, PctPrivateCoverageAlone,
    and the rest of them with some form of negative or positive skewness in data distribution'''


''' 3. There are lots of missing data in all the features, each having at least 401 missing values.
    On the average, this amounts to about 13% of missing data which is very huge
    Each row having at least one missing value except one row (Sample #2012).
    The highest number of missing value in a row is 13 and the lowest 0.
    There are neither duplicate rows nor columns'''


''' 4.We also observe anormalous duplicate values in features 'avgAnncount' and 'incidenceRate'
    with 2096 and 1579 duplicate values
    (1962.667684, 453.549422) respectively. Not replacing them with NaN will affect our model'''


''' 5. With the skewness in the data distribution of the majority of the features,
    it will be difficult to clean this data by any of the available methods (padding,
    backfill or linear extrapolation method).
    Not even the default dropna method (that deletes rows having at least one non-null)
    can be implemented because just at a glance (from data in the first 50 rows),
    nearly or all the rows in this data have at least 1 missing value (1 True).
    However, we can use the dropna method to remove rows based on certain conditions
    (For example, dropping rows with at least 10 missing values).
    This data is super flawed, too much missing information.'''
```

Out[337…

```
' 5. With the skewness in the data distribution of the majority of the features, \n    it will be difficult to clean this data by any of the available methods (padding, \n    backfill or linear extrapolation method). \n
```

```
Not even the default dropna method (that deletes rows having at least one non-null) \n    can be implemented be
cause just at a glance (from data in the first 50 rows), \n    nearly or all the rows in this data have at leas
t 1 missing value (1 True). \n    However, we can use the dropna method to remove rows based on certain conditi
ons \n    (For example, dropping rows with at least 10 missing values). \n    This data is super flawed, too mu
ch missing information.'
```

In [338…

```
df.isnull().any(axis = 1).sum()
```

Out[338…

```
3046
```

In [339…

```
df.dropna(subset=['avgAnnCount', 'avgDeathsPerYear', 'TARGET_deathRate',
       'incidenceRate', 'medIncome', 'popEst2015', 'povertyPercent',
       'studyPerCap', 'binnedInc', 'MedianAge', 'MedianAgeMale',
       'MedianAgeFemale', 'Geography', 'AvgHouseholdSize', 'PercentMarried',
       'PctNoHS18_24', 'PctHS18_24', 'PctSomeCol18_24', 'PctBachDeg18_24',
       'PctHS25_Over', 'PctBachDeg25_Over', 'PctEmployed16_Over',
       'PctUnemployed16_Over', 'PctPrivateCoverage', 'PctPrivateCoverageAlone',
       'PctEmpPrivCoverage', 'PctPublicCoverage', 'PctPublicCoverageAlone',
       'PctWhite', 'PctBlack', 'PctAsian', 'PctOtherRace',
       'PctMarriedHouseholds', 'BirthRate'])
```

Out[339…

	Unnamed: 0	avgAnnCount	avgDeathsPerYear	TARGET_deathRate	incidenceRate	medIncome	popEst2015	povertyPercent	studyI
2012	2012	181.0	70.0	188.8	517.2	42088.0	28880.0	18.0	

In [340…

```
df.describe()
```

Out[340…

	Unnamed: 0	avgAnnCount	avgDeathsPerYear	TARGET_deathRate	incidenceRate	medIncome	popEst2015	povertyPercer
count	3047.000000	2637.000000	2638.000000	2629.000000	2640.000000	2633.000000	2.630000e+03	2633.00000
mean	1523.000000	616.813493	190.055724	178.748840	447.960586	47024.274592	1.023032e+05	16.93820
std	879.737461	1467.543751	527.267114	27.803006	55.406582	11921.226519	3.389258e+05	6.46466
min	0.000000	7.000000	3.000000	59.700000	201.300000	22640.000000	8.290000e+02	3.20000
25%	761.500000	77.000000	27.000000	161.300000	419.500000	38989.000000	1.155525e+04	12.20000
50%	1523.000000	173.000000	60.000000	178.100000	453.549422	45235.000000	2.697150e+04	15.90000

	Unnamed: 0	avgAnnCount	avgDeathsPerYear	TARGET_deathRate	incidenceRate	medIncome	popEst2015	povertyPercer
75%	2284.500000	526.000000	148.000000	195.200000	480.900000	52513.000000	6.860925e+04	20.50000
max	3046.000000	38150.000000	14010.000000	362.800000	1206.900000	125635.000000	1.017029e+07	47.40000

In [341…

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3047 entries, 0 to 3046
Data columns (total 35 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Unnamed: 0               3047 non-null   int64
 1   avgAnnCount              2637 non-null   float64
 2   avgDeathsPerYear         2638 non-null   float64
 3   TARGET_deathRate         2629 non-null   float64
 4   incidenceRate            2640 non-null   float64
 5   medIncome                2633 non-null   float64
 6   popEst2015               2630 non-null   float64
 7   povertyPercent           2633 non-null   float64
 8   studyPerCap              2644 non-null   float64
 9   binnedInc                2632 non-null   object
 10  MedianAge                2634 non-null   float64
 11  MedianAgeMale            2636 non-null   float64
 12  MedianAgeFemale          2631 non-null   float64
 13  Geography                2641 non-null   object
 14  AvgHouseholdSize         2638 non-null   float64
 15  PercentMarried           2639 non-null   float64
 16  PctNoHS18_24             2640 non-null   float64
 17  PctHS18_24               2634 non-null   float64
 18  PctSomeCol18_24          653 non-null    float64
 19  PctBachDeg18_24          2634 non-null   float64
 20  PctHS25_Over             2631 non-null   float64
 21  PctBachDeg25_Over        2638 non-null   float64
 22  PctEmployed16_Over       2497 non-null   float64
 23  PctUnemployed16_Over     2641 non-null   float64
 24  PctPrivateCoverage       2639 non-null   float64
 25  PctPrivateCoverageAlone  2118 non-null   float64
 26  PctEmpPrivCoverage       2636 non-null   float64
 27  PctPublicCoverage        2639 non-null   float64
 28  PctPublicCoverageAlone   2646 non-null   float64
 29  PctWhite                 2637 non-null   float64
 30  PctBlack                 2632 non-null   float64
```

```
 31  PctAsian                2634 non-null   float64
 32  PctOtherRace            2630 non-null   float64
 33  PctMarriedHouseholds    2637 non-null   float64
 34  BirthRate               2634 non-null   float64
dtypes: float64(32), int64(1), object(2)
memory usage: 833.3+ KB
```

In [342…

```
df.isnull().sum(axis=1).max()
```

Out[342…

```
13
```

In [343…

```
df.isnull().sum(axis=1).min()
```

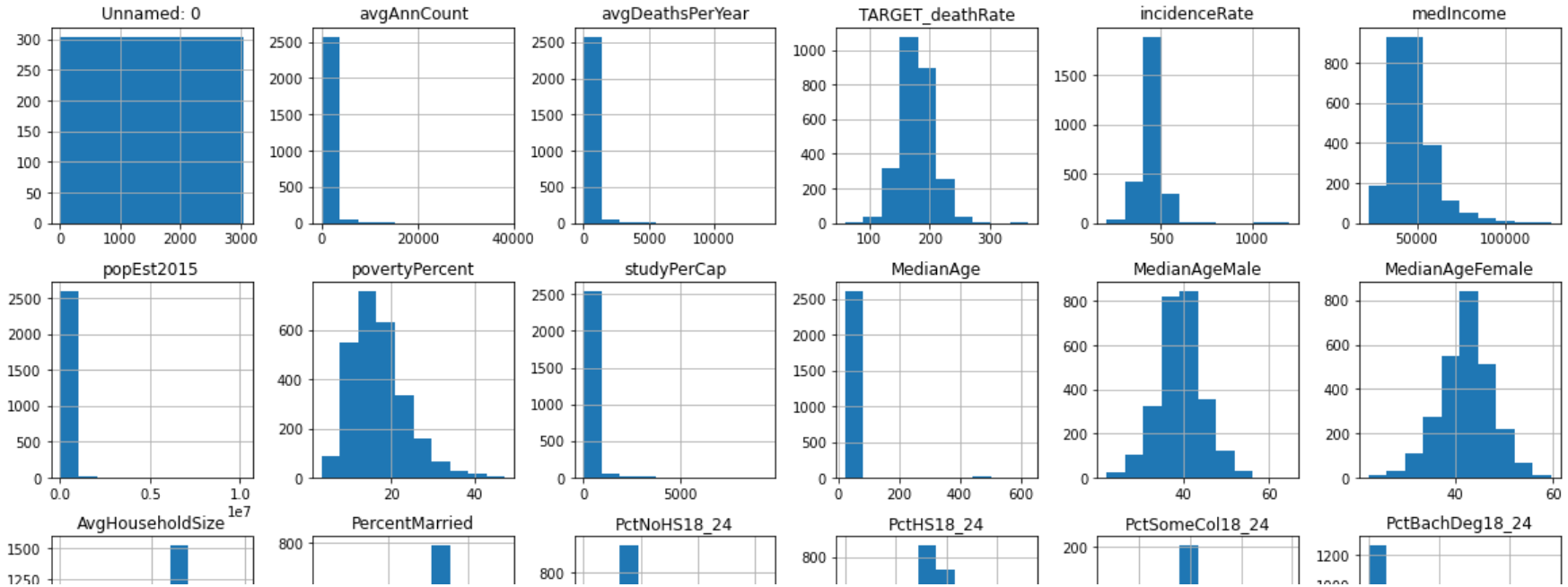
Out[343…

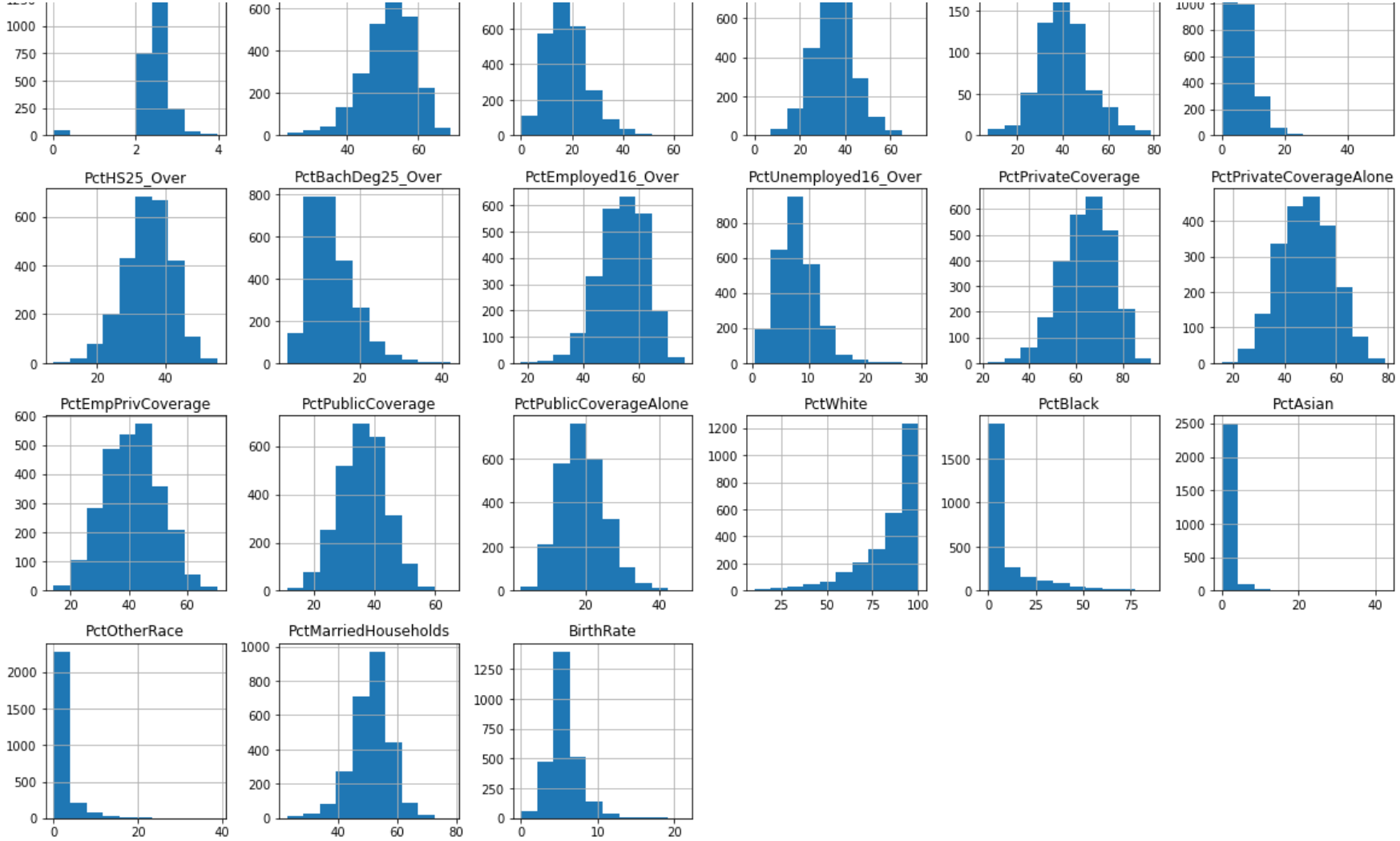
```
0
```

In [344…

```
from matplotlib import pyplot


df.hist()
pyplot.gcf().set_size_inches(20,20)
pyplot.show()
```





In [ ]:

In [345…

```
#Question 3

# Identify and report missing values and duplicate samples in the data

''' 1. In each of these columns, there are at least 401 missing values'''


''' 2. We also notice duplicate values with abnormal decimal places.
```

```
    Features 'avgAnncount' and 'incidenceRate'
    with 2096 and 1579 duplicate values (1962.667684, 453.549422) respectively.'''

''' 3. There are neither duplicate rows nor columns'''
```

Out[345…

```
' 3. There are neither duplicate rows nor columns'
```

In [346…

```
df.isna().sum()
```

Out[346…

```
Unnamed: 0                  0
avgAnnCount               410
avgDeathsPerYear          409
TARGET_deathRate          418
incidenceRate             407
medIncome                 414
popEst2015                417
povertyPercent            414
studyPerCap               403
binnedInc                 415
MedianAge                 413
MedianAgeMale             411
MedianAgeFemale           416
Geography                 406
AvgHouseholdSize          409
PercentMarried            408
PctNoHS18_24              407
PctHS18_24                413
PctSomeCol18_24          2394
PctBachDeg18_24           413
PctHS25_Over              416
PctBachDeg25_Over         409
PctEmployed16_Over        550
PctUnemployed16_Over      406
PctPrivateCoverage        408
PctPrivateCoverageAlone   929
PctEmpPrivCoverage        411
PctPublicCoverage         408
PctPublicCoverageAlone    401
PctWhite                  410
PctBlack                  415
PctAsian                  413
PctOtherRace              417
PctMarriedHouseholds      410
```

```
BirthRate                  413
dtype: int64
```

In [ ]:

In [347…

```
#Question 4

#Clean the data of missing values and also of duplicate values.
#(Explain your choices and give justifications for any method you use to clean the data of missing values)


''' 1. First off, we need to drop rows that have all values missing. since we do not have any of such,
    we can then proceed to dropping rows with at least "n" number of missing values.
    In this data file, the highest number of missing data in a row is 13 and the lowest is 0
    Furthermore, if we do not have information about the (AvgAnnCount and AvgDeathsPerYear),
    (TARGET_DeathRate and incidenceRate)
    (which are one of the most important features in this data)
    then the other information in the columns can as well be rendered useless and can be dropped.
    Dropping such rows using the "dropna" method will be very usefull here.
    A total of 125 rows were removed which amounts to a removal of about 4.1% of the whole data.
    These features appear highly dependent on each other,
    so I then derive a dividing/multiplying factor which can be useful in replacing missing values
    in these columns.

    We also observe anormalous duplicate values in features
    'avgAnncount' and 'incidenceRate' with 2096 and 1579 duplicate values
    (1962.667684, 453.549422) respectively. Not replacing them with NaN will affect our model
    (Please NOTE: For some reason unknown to me, I am unable to replace the latter values in
    column 5 ('incidenceRate') with NaN).
    By replacing 1962.667684 with NaN and the finding the mean in column 2 ('avgAnncount') and in
    column 3 ('AvgDeathsPerYear'), .
    A dividing/multiplying factor is derived by which we use to
    replace missing values in these 4 columns mentioned above'''

''' 2. Furthermore, If the highest number of missing values in a row is 13, I think it is unreasonable
    to Drop rows with at least 10 missing values
    (dropping a total of another 92 rows (dropping about 7.1% of our data)).
    It is unreasonable because we have a  meagre amount of samples(a few thousands compared to millions)'''

''' 3. For features in 'AvgAnnCount' and AvgDeathsPerYear,
    we can remove the rows that have missing values (on the same row) in these two features
    since they are dependent on each other.
    If we have missing values for both features, then the study in question can be considered useless. '''
```

```
''' 4. There are neither duplicate columns nor rows. However, in the last few rows of two features namely:
    AvgAnnCount and incidenceRate both happen to contain duplicate anomalous float values
    which appear different from the rest of the values respectively.
    The 'AvgDeathsPerYear' features have a relationship with features of the 'AvgAnnCount' column
    likewise for 'TARGET_deathRate' and 'incidenceRate'.
    We only need to calculate the factor with a simple mathematical expression as shown below'''

''' 5. For the following features below, we can replace missing values by using the median of the values
       'medIncome', 'popEst2015', 'povertyPercent', 'binnedInc',
       'MedianAge', 'MedianAgeMale', 'MedianAgeFemale', 'Geography',
       'PercentMarried', 'PctNoHS18_24', 'PctHS18_24',
       'PctSomeCol18_24', 'PctBachDeg18_24', 'PctHS25_Over',
       'PctBachDeg25_Over', 'PctEmployed16_Over', 'PctUnemployed16_Over',
       'PctPrivateCoverageAlone', 'PctEmpPrivCoverage', 'PctBlack',
       'PctAsian', 'PctOtherRace', 'PctMarriedHouseholds', 'BirthRate'.
    The decision to fill up missing values is based on the fact that
    the minimum and maximum values are so wide apart
    and also because the average standard deviation of the values
    from the mean in these features is also very high. The density curves are non-gaussian (skewed)'''

''' 6. For features in studyPerCap, (2074(mode) - 401(number of missing values)
    number of rows have a value of 0.000000 which amounts to about 63% of the whole data in this feature.
    I really think it's best to replace this 401 missing values with the mode (0.0000000).
    Additionally, if we choose to use the median, this also gives us a value of 0.000000 '''

''' 7. For the rest of the features like 'avgHouseholdsize', 'pctPrivateCoverage',
    we replace the missing values with the mean because
    the values have a lower standard deviation from the mean
    and because the density curve is normalized though not very perfect (density curve is gaussian)'''

''' 8. For features of object type like the 'binnedInc' and 'Geography',
    we replace missing values by mode and padding method respectively. For the former,
    there is a mode while for the latter,
    there is neither a mode, mean nor median and padding is the best method
    for replacing missing values of this data type'''
```

Out[347…

```
" 8. For features of object type like the 'binnedInc' and 'Geography', \n    we replace missing values by mode and padding method respectively. For the former, \n    there is a mode while for the latter, \n    there is neither a mode, mean nor median and padding is the best method \n    for replacing missing values of this data type"
```

In [348…

```
import numpy as np
df.replace(to_replace = 1962.667684, value = np.nan, inplace = True)
```

In [349…
```
df.dropna(subset=['avgAnnCount', 'avgDeathsPerYear'], how = 'all', inplace = True)
```

In [350…
```
df
```

Out[350…

	Unnamed: 0	avgAnnCount	avgDeathsPerYear	TARGET_deathRate	incidenceRate	medIncome	popEst2015	povertyPercent	study
0	0	1397.0	469.0	164.9	489.800000	61898.0	260131.0	11.2	499.
1	1	173.0	70.0	161.3	411.600000	48127.0	NaN	18.6	23
2	2	NaN	50.0	174.7	349.700000	49348.0	21026.0	14.6	47.
3	3	427.0	NaN	194.8	430.400000	44243.0	75882.0	NaN	342.
4	4	57.0	NaN	144.4	350.100000	49955.0	10321.0	12.5	0.
...	...	...	...	...	...	...	...	...	
3042	3042	NaN	15.0	NaN	453.549422	46961.0	6343.0	12.4	0.
3043	3043	NaN	43.0	150.1	453.549422	48609.0	37118.0	NaN	377
3044	3044	NaN	46.0	153.9	453.549422	NaN	NaN	15.0	1968.
3045	3045	NaN	52.0	175.0	453.549422	50745.0	25609.0	13.3	0.
3046	3046	NaN	48.0	213.6	453.549422	41193.0	37030.0	13.9	0.

2976 rows × 35 columns

In [351…
```
df.dropna(subset=['TARGET_deathRate', 'incidenceRate'], how = 'all', inplace = True)
```

In [352…
```
df
```

Out[352…

	Unnamed: 0	avgAnnCount	avgDeathsPerYear	TARGET_deathRate	incidenceRate	medIncome	popEst2015	povertyPercent	study
0	0	1397.0	469.0	164.9	489.800000	61898.0	260131.0	11.2	499.
1	1	173.0	70.0	161.3	411.600000	48127.0	NaN	18.6	23
2	2	NaN	50.0	174.7	349.700000	49348.0	21026.0	14.6	47.
3	3	427.0	NaN	194.8	430.400000	44243.0	75882.0	NaN	342.
4	4	57.0	NaN	144.4	350.100000	49955.0	10321.0	12.5	0.
...	...	...	...	...	...	...	...	...	
3042	3042	NaN	15.0	NaN	453.549422	46961.0	6343.0	12.4	0.
3043	3043	NaN	43.0	150.1	453.549422	48609.0	37118.0	NaN	377
3044	3044	NaN	46.0	153.9	453.549422	NaN	NaN	15.0	1968.
3045	3045	NaN	52.0	175.0	453.549422	50745.0	25609.0	13.3	0.

	Unnamed: 0	avgAnnCount	avgDeathsPerYear	TARGET_deathRate	incidenceRate	medIncome	popEst2015	povertyPercent	study
3046	3046	NaN	48.0	213.6	453.549422	41193.0	37030.0	13.9	0.

2922 rows × 35 columns

In [353…

```
df.columns
```

Out[353…

```
Index(['Unnamed: 0', 'avgAnnCount', 'avgDeathsPerYear', 'TARGET_deathRate',
       'incidenceRate', 'medIncome', 'popEst2015', 'povertyPercent',
       'studyPerCap', 'binnedInc', 'MedianAge', 'MedianAgeMale',
       'MedianAgeFemale', 'Geography', 'AvgHouseholdSize', 'PercentMarried',
       'PctNoHS18_24', 'PctHS18_24', 'PctSomeCol18_24', 'PctBachDeg18_24',
       'PctHS25_Over', 'PctBachDeg25_Over', 'PctEmployed16_Over',
       'PctUnemployed16_Over', 'PctPrivateCoverage', 'PctPrivateCoverageAlone',
       'PctEmpPrivCoverage', 'PctPublicCoverage', 'PctPublicCoverageAlone',
       'PctWhite', 'PctBlack', 'PctAsian', 'PctOtherRace',
       'PctMarriedHouseholds', 'BirthRate'],
      dtype='object')
```

In [354…

```
data35 = df['BirthRate']
data34 = df['PctMarriedHouseholds']
data33 = df['PctOtherRace']
data32 = df['PctAsian']
data31 = df['PctBlack']
data30 = df['PctWhite']
data29 = df['PctPublicCoverageAlone']
data28 = df['PctPublicCoverage']
data27 = df['PctEmpPrivCoverage']
data26 = df['PctPrivateCoverageAlone']
data25 = df['PctPrivateCoverage']
data24 = df['PctUnemployed16_Over']
data23 = df['PctEmployed16_Over']
data22 = df['PctBachDeg25_Over']
data21 = df['PctHS25_Over']
data20 = df['PctBachDeg18_24']
data19 = df['PctSomeCol18_24']
data18 = df['PctHS18_24']
data17 = df['PctNoHS18_24']
data16= df['PercentMarried']
```

```
data15= df['AvgHouseholdSize']
data14= df['Geography']
data13= df['MedianAgeFemale']
data12 = df['MedianAgeMale']
data10 =df['binnedInc']
data11= df['MedianAge']
data9= df['studyPerCap']
data8= df['povertyPercent']
data7= df['popEst2015']
data6= df['medIncome']
data5= df['incidenceRate']
data4= df['TARGET_deathRate']
data3= df['avgDeathsPerYear']
data2= df['avgAnnCount']
```

In [355…

```
import numpy as np
data2.replace(to_replace = 1962.667684, value = np.nan).mean()
```

Out[355… 522.7756622516556

In [356…

```
data3.mean()
```

Out[356… 191.11978361669242

In [357…

```
data4.mean()
```

Out[357… 178.84869395711513

In [358…

```
data5.mean()
```

Out[358… 447.87964326605726

In [359…

```
factorofD2AndD3 =522.78/191.12
factorofD2AndD3
```

Out[359… 2.7353495186270402

In [360…

```
relOfIRAndTargetDR = 447.88/178.85
R = relOfIRAndTargetDR

factor2 = data2/2.74
factor3 = data3 *2.74
factor4 = data5/R
factor5 = data4*R

data2.fillna(factor3, inplace = True)
data3.fillna(factor2, inplace = True)
data4.fillna(factor4, inplace = True)
data5.fillna(factor5, inplace = True)
df
```

Out[360…

	Unnamed: 0	avgAnnCount	avgDeathsPerYear	TARGET_deathRate	incidenceRate	medIncome	popEst2015	povertyPercent	study
0	0	1397.00	469.000000	164.900000	489.800000	61898.0	260131.0	11.2	499.
1	1	173.00	70.000000	161.300000	411.600000	48127.0	NaN	18.6	23
2	2	137.00	50.000000	174.700000	349.700000	49348.0	21026.0	14.6	47.
3	3	427.00	155.839416	194.800000	430.400000	44243.0	75882.0	NaN	342.
4	4	57.00	20.802920	144.400000	350.100000	49955.0	10321.0	12.5	0.
…	…	…	…	…	…	…	…	…	
3042	3042	41.10	15.000000	181.113946	453.549422	46961.0	6343.0	12.4	0.
3043	3043	117.82	43.000000	150.100000	453.549422	48609.0	37118.0	NaN	377
3044	3044	126.04	46.000000	153.900000	453.549422	NaN	NaN	15.0	1968.

	Unnamed: 0	avgAnnCount	avgDeathsPerYear	TARGET_deathRate	incidenceRate	medIncome	popEst2015	povertyPercent	study
3045	3045	142.48	52.000000	175.000000	453.549422	50745.0	25609.0	13.3	0.
3046	3046	131.52	48.000000	213.600000	453.549422	41193.0	37030.0	13.9	0.

2922 rows × 35 columns

In [361…

```
df_x35 = data35.median()
data35.fillna(df_x35, inplace = True)

df_x34 = data34.median()
data34.fillna(df_x34, inplace = True)

df_x33 = data33.median()
data33.fillna(df_x33, inplace = True)

df_x32 = data32.median()
data32.fillna(df_x32, inplace = True)

df_x31 = data31.median()
data31.fillna(df_x31, inplace = True)

df_x30 = data30.mean()
data30.fillna(df_x30, inplace = True)

df_x29 = data29.median()
data29.fillna(df_x29, inplace = True)

df_x28 = data28.median()
data28.fillna(df_x28, inplace = True)

df_x27 = data27.median()
data27.fillna(df_x27, inplace = True)

df_x26 = data26.median()
data26.fillna(df_x26, inplace = True)

df_x25 = data25.median()
data25.fillna(df_x25, inplace = True)
```

```
df_x24 = data24.median()
data24.fillna(df_x24, inplace = True)

df_x23 = data23.median()
data23.fillna(df_x35, inplace = True)

df_x22 = data22.median()
data22.fillna(df_x22, inplace = True)

df_x21 = data21.median()
data21.fillna(df_x21, inplace = True)

df_x20 = data20.median()
data20.fillna(df_x20, inplace = True)

df_x19 = data19.median()
data19.fillna(df_x19, inplace = True)

df_x18 = data18.mean()
data18.fillna(df_x18, inplace = True)

df_x17 = data17.median()
data17.fillna(df_x17, inplace = True)

df_x16 = data16.median()
data16.fillna(df_x16, inplace = True)

df_x15 = data15.median()
data15.fillna(df_x15, inplace = True)

data14.fillna(method = 'pad', inplace = True)

df_x13 = data13.median()
data13.fillna(df_x13, inplace = True)

df_x12 = data12.median()
data12.fillna(df_x12, inplace = True)

df_x11 = data11.median()
data11.fillna(df_x11, inplace = True)

df_x10 = data10.mode()[0]
data10.fillna(df_x10, inplace = True)
```

```
df_x9 = data9.median()
data9.fillna(df_x9, inplace = True)

df_x8 = data8.median()
data8.fillna(df_x8, inplace = True)

df_x7 = data7.median()
data7.fillna(df_x7, inplace = True)

df_x6 = data6.median()
data6.fillna(df_x6, inplace = True)
```

In [362…

```
#Question5
#Report the structure of the final data after cleaning (number of samples and number of features)

'''1. At the end of the data cleaning process, I report a total of 2,922 samples with 35 features'''
```

Out[362… '1. At the end of the data cleaning process, I report a total of 2,922 samples with 35 features'

In [363…

```
newDf = df
```

In [365…

```
newDf
```

Out[365…

	Unnamed: 0	avgAnnCount	avgDeathsPerYear	TARGET_deathRate	incidenceRate	medIncome	popEst2015	povertyPercent	study
0	0	1397.00	469.000000	164.900000	489.800000	61898.0	260131.0	11.2	499.
1	1	173.00	70.000000	161.300000	411.600000	48127.0	27157.0	18.6	23
2	2	137.00	50.000000	174.700000	349.700000	49348.0	21026.0	14.6	47.
3	3	427.00	155.839416	194.800000	430.400000	44243.0	75882.0	15.9	342.

	Unnamed: 0	avgAnnCount	avgDeathsPerYear	TARGET_deathRate	incidenceRate	medIncome	popEst2015	povertyPercent	study
4	4	57.00	20.802920	144.400000	350.100000	49955.0	10321.0	12.5	0.
...	...	...	...	...	...	...	...	...	
3042	3042	41.10	15.000000	181.113946	453.549422	46961.0	6343.0	12.4	0.
3043	3043	117.82	43.000000	150.100000	453.549422	48609.0	37118.0	15.9	377
3044	3044	126.04	46.000000	153.900000	453.549422	45209.0	27157.0	15.0	1968.
3045	3045	142.48	52.000000	175.000000	453.549422	50745.0	25609.0	13.3	0.
3046	3046	131.52	48.000000	213.600000	453.549422	41193.0	37030.0	13.9	0.

2922 rows × 35 columns

In [366…

```
newDf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2922 entries, 0 to 3046
Data columns (total 35 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Unnamed: 0              2922 non-null   int64
 1   avgAnnCount             2922 non-null   float64
 2   avgDeathsPerYear        2922 non-null   float64
 3   TARGET_deathRate        2922 non-null   float64
 4   incidenceRate           2922 non-null   float64
 5   medIncome               2922 non-null   float64
 6   popEst2015              2922 non-null   float64
 7   povertyPercent          2922 non-null   float64
```

```
 8   studyPerCap              2922 non-null   float64
 9   binnedInc                2922 non-null   object
 10  MedianAge                2922 non-null   float64
 11  MedianAgeMale            2922 non-null   float64
 12  MedianAgeFemale          2922 non-null   float64
 13  Geography                2922 non-null   object
 14  AvgHouseholdSize         2922 non-null   float64
 15  PercentMarried           2922 non-null   float64
 16  PctNoHS18_24             2922 non-null   float64
 17  PctHS18_24               2922 non-null   float64
 18  PctSomeCol18_24          2922 non-null   float64
 19  PctBachDeg18_24          2922 non-null   float64
 20  PctHS25_Over             2922 non-null   float64
 21  PctBachDeg25_Over        2922 non-null   float64
 22  PctEmployed16_Over       2922 non-null   float64
 23  PctUnemployed16_Over     2922 non-null   float64
 24  PctPrivateCoverage       2922 non-null   float64
 25  PctPrivateCoverageAlone  2922 non-null   float64
 26  PctEmpPrivCoverage       2922 non-null   float64
 27  PctPublicCoverage        2922 non-null   float64
 28  PctPublicCoverageAlone   2922 non-null   float64
 29  PctWhite                 2922 non-null   float64
 30  PctBlack                 2922 non-null   float64
 31  PctAsian                 2922 non-null   float64
 32  PctOtherRace             2922 non-null   float64
 33  PctMarriedHouseholds     2922 non-null   float64
 34  BirthRate                2922 non-null   float64
dtypes: float64(32), int64(1), object(2)
memory usage: 821.8+ KB
```

In [367…

```
newDf.describe()
```

Out[367…

	Unnamed: 0	avgAnnCount	avgDeathsPerYear	TARGET_deathRate	incidenceRate	medIncome	popEst2015	povertyPercer
count	2922.000000	2922.000000	2922.000000	2922.000000	2922.000000	2922.000000	2.922000e+03	2922.00000
mean	1517.783710	499.596516	187.075186	178.577147	447.972800	46804.337782	9.328785e+04	16.78617
std	878.262247	1403.330397	511.705197	27.176822	57.903664	11162.234501	3.210949e+05	6.0107(
min	0.000000	7.000000	2.919708	59.700000	149.502018	22640.000000	8.290000e+02	3.20000
25%	756.250000	69.000000	28.000000	161.900000	416.925000	40122.000000	1.370300e+04	12.80000
50%	1517.500000	150.000000	60.583942	178.600000	453.549422	45209.000000	2.715700e+04	15.90000

	Unnamed: 0	avgAnnCount	avgDeathsPerYear	TARGET_deathRate	incidenceRate	medIncome	popEst2015	povertyPercer
75%	2277.750000	383.000000	148.000000	194.600000	482.547600	51342.250000	5.765750e+04	19.60000
max	3046.000000	38150.000000	14010.000000	362.800000	1206.900000	125635.000000	1.017029e+07	47.40000

In [368…

```
newDf.isna().sum()
```

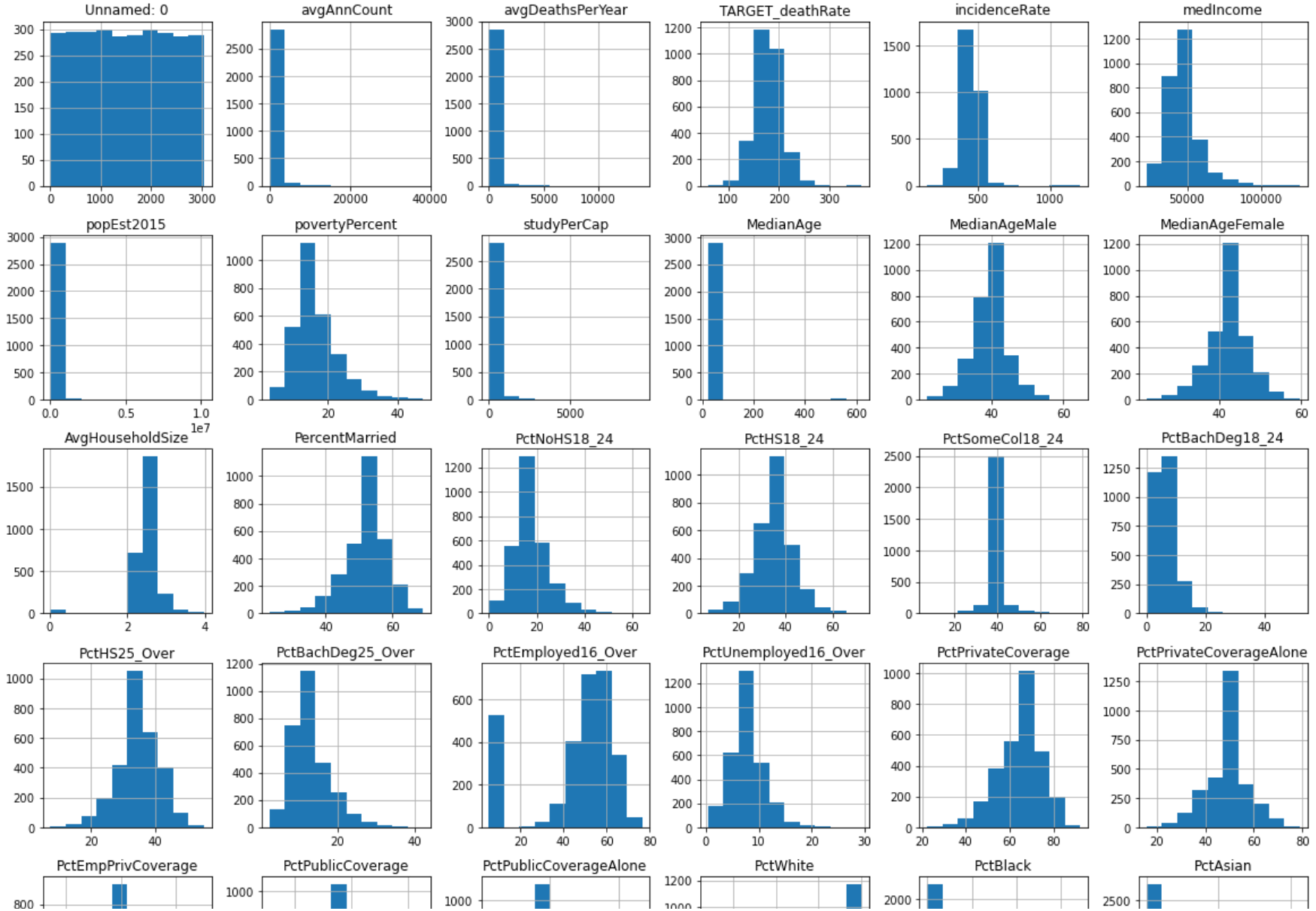
Out[368…

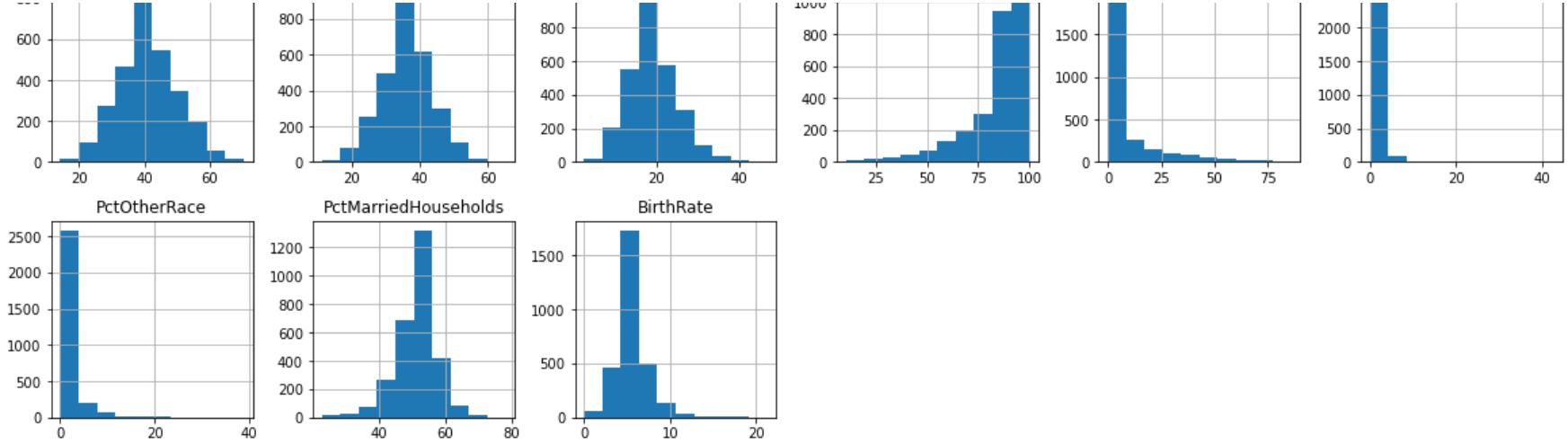
```
Unnamed: 0                0
avgAnnCount               0
avgDeathsPerYear          0
TARGET_deathRate          0
incidenceRate             0
medIncome                 0
popEst2015                0
povertyPercent            0
studyPerCap               0
binnedInc                 0
MedianAge                 0
MedianAgeMale             0
MedianAgeFemale           0
Geography                 0
AvgHouseholdSize          0
PercentMarried            0
PctNoHS18_24              0
PctHS18_24                0
PctSomeCol18_24           0
PctBachDeg18_24           0
PctHS25_Over              0
PctBachDeg25_Over         0
PctEmployed16_Over        0
PctUnemployed16_Over      0
PctPrivateCoverage        0
PctPrivateCoverageAlone   0
PctEmpPrivCoverage        0
PctPublicCoverage         0
PctPublicCoverageAlone    0
PctWhite                  0
PctBlack                  0
PctAsian                  0
PctOtherRace              0
PctMarriedHouseholds      0
BirthRate                 0
dtype: int64
```

In [369…

```
from matplotlib import pyplot


newDf.hist()
pyplot.gcf().set_size_inches(20,20)
pyplot.show()
```





In [371…
```
newDf.to_csv('cancerdata_cleaned.csv')
```

In [376…
```
newDf.to_csv('/Users/tomisin/Dropbox/My Mac (Tomisins-MacBook-Pro.local)/Documents/MY WORKSPACE/Database/cancer
```

In [ ]: