DataEng: Data Integration Activity

This week you will gain hands-on experience with Data Integration by combining data from two distinct sources into a unified DataFrame for analysis.

Submit: Make a copy of this document and use it to record your results. Store a PDF copy of the document in your git repository along with any needed code before submitting for this week.

Your job is to integrate **county-level COVID-19 data** with the **ACS Census Tract data for 2017** to build a model that allows you to relate COVID numbers with economic data such as population, per capita income and poverty level. To do this you should build a pandas DataFrame that has a row per USA county (there are more than 3000 counties in the USA) and includes the following columns:

County - name of the county
State - name of the state in which the county resides
TotalCases - total number of COVID cases for this county as of February 20, 2021
Dec2020Cases - number of COVID cases recorded in this county in December of 2020
TotalDeaths - total number of COVID deaths for this county as of February 20, 2021
Dec2020Deaths - number of COVID deaths recorded in this county in December of 2020
Population - population of this county
Poverty - % of people in poverty in this county
PerCapitaIncome - per capita personal income for this county

We hope that you make it all the way through to the end. Regardless, use your time wisely to gain python programming experience and learn as much as you can about building integrated multi-source data models using python and pandas.

For this activity you should use whichever environment is convenient for you to develop with python 3 and pandas. You are not required to use GCP, but you can use it if you prefer.

Submit: In-class Activity Submission Form

A. Aggregate Census Data to County Level

Your integration will use two different dimensions: location (as indicated by state and county) and time. You should greatly simplify your processing and reduce your time by pre-processing your data along each of these dimensions.

The ACS data is separated into “Census Tracts” which are regions within counties that correspond to groups of approximately 4000 people. The Census Bureau defines these

to help organize the actual job of collecting census data, but this grouping can make your Data Engineering job more more challenging. This level of detail is not needed for your county-level analysis, and you can greatly decrease your efforts by aggregating per-tract data to the county level.

Create a python program that produces a one-row-per-county version of the ACS data set. To do this you will need to think about how to properly aggregate Census Tract-level data into County-level summaries.

In this step you can also eliminate unneeded columns from the ACS data.

Question: Show your aggregated county-level data rows for the following counties: Loudon County Virginia, Washington County Oregon, Harlan County Kentucky, Malheur County Oregon

```
TotalPop      sum     3.745580e+05
IncomePerCap  sum     3.225025e+06
Poverty       mean    3.884375e+00
Name: (Loudoun County, Virginia), dtype: float64

TotalPop      sum     5.720710e+05
IncomePerCap  sum     3.636965e+06
Poverty       mean    1.044615e+01
Name: (Washington County, Oregon), dtype: float64

TotalPop      sum      27548.000000
IncomePerCap  sum     176114.000000
Poverty       mean       33.318182
Name: (Harlan County, Kentucky), dtype: float64

TotalPop      sum      30421.000000
IncomePerCap  sum     125765.000000
Poverty       mean       24.414286
Name: (Malheur County, Oregon), dtype: float64
```

B. Simplify the COVID Data

You can simplify the COVID data along the time dimension. The COVID data set contains day-level resolution data from (approximately) March of 2020 through February of 2021. However, you will only need four data points per county: total cases, total deaths, cases reported during December of 2020 and deaths reported during December 2020.

Create a python program that reduces the COVID data to one line per county.

Question: Show your simplified COVID data for the counties listed above.

```
cases          2496450.0
deaths           35820.0
Dec2020Cases    376223.0
Dec2020Deaths     4729.0
Name: (Loudoun, Virginia), dtype: float64

cases          2157339.0
deaths           22455.0
Dec2020Cases    424620.0
Dec2020Deaths     3860.0
Name: (Washington, Oregon), dtype: float64

cases          205984.0
deaths           3994.0
Dec2020Cases    38959.0
Dec2020Deaths     506.0
Name: (Harlan, Kentucky), dtype: float64

cases          453634.0
deaths           7770.0
Dec2020Cases    82916.0
Dec2020Deaths    1465.0
Name: (Malheur, Oregon), dtype: float64
```

C. Integrate COVID Data with ACS Data

Create a single pandas DataFrame containing one row per county and using the columns described above. You are free to add additional columns if needed. For example, you might want to normalize all of the COVID data by the population of each county so that you have a consistent “number of cases/deaths per 100000 residents” value for each county.

Question: List your integrated data for all counties in the State of Oregon.

TotalPop, sum)	(IncomePerCap, sum)	(Poverty, mean)	cases	deaths	Dec2020Cases	Dec2020Deaths

county							
Baker	15980	154241.0	15.000000	55586.0	663.0	11688.0	133.0
Benton	88249	538668.0	23.644444	180225.0	2304.0	34260.0	278.0
Clackamas	399962	3000217.0	9.320000	1284402.0	20040.0	261810.0	3125.0
Clatsop	38021	311931.0	12.481818	77666.0	287.0	14439.0	47.0
Columbia	50207	281097.0	12.530000	105324.0	1363.0	21459.0	266.0
Coos	62921	344348.0	17.415385	100097.0	969.0	18806.0	151.0
Crook	21717	95834.0	15.050000	55863.0	1134.0	11048.0	196.0
Curry	22377	134496.0	16.300000	30045.0	393.0	6741.0	72.0
Deschutes	175321	764025.0	12.208333	509974.0	4141.0	102490.0	563.0
Douglas	107576	554588.0	16.731818	174952.0	3983.0	37590.0	964.0
Gilliam	1910	24178.0	9.900000	4691.0	76.0	898.0	25.0
Grant	7209	47710.0	15.850000	18551.0	94.0	4895.0	31.0
Harney	7195	50349.0	16.300000	17024.0	291.0	3717.0	34.0
Hood	22938	116712.0	12.150000	NaN	NaN	NaN	NaN

Jackson	212070	1120480.0	17.882927	713288.0	7221.0	154535.0	1655.0
Jefferson	22707	136138.0	20.316667	200346.0	2630.0	36278.0	409.0
Josephine	84514	386865.0	19.131250	153675.0	2638.0	27180.0	407.0
Klamath	66018	474248.0	18.930000	224256.0	2857.0	45118.0	373.0
Lake	7807	42243.0	19.200000	25357.0	348.0	5358.0	76.0
Lane	363471	2368975.0	18.529070	850956.0	10372.0	178816.0	2215.0
Lincoln	47307	455726.0	17.623529	153979.0	3117.0	24041.0	502.0
Linn	121074	513507.0	16.923810	324636.0	5949.0	66702.0	891.0
Malheur	30421	125765.0	24.414286	453634.0	7770.0	82916.0	1465.0
Marion	330453	1502424.0	15.329310	1974030.0	34089.0	365801.0	5720.0
Morrow	11153	46343.0	13.450000	139209.0	1447.0	23219.0	227.0
Multnomah	788459	6245725.0	15.730588	3374737.0	58787.0	680418.0	10244.0
Polk	79666	295607.0	18.641667	268036.0	5480.0	50986.0	743.0
Sherman	1635	34226.0	13.700000	5807.0	0.0	855.0	0.0

Tillamook	25840	206446.0	15.437500	34370.0	92.0	6850.0	0.0
Umatilla	76736	348007.0	16.520000	933975.0	10661.0	154995.0	1645.0
Union	25810	212071.0	17.425000	161223.0	1533.0	28227.0	338.0
Wallowa	6864	80829.0	14.400000	13017.0	449.0	2306.0	93.0
Wasco	25687	200718.0	13.037500	121202.0	3039.0	22511.0	621.0
Washington	572071	3636965.0	10.446154	2157339.0	22455.0	424620.0	3860.0
Wheeler	1415	21268.0	20.600000	1454.0	53.0	359.0	2.0
Yamhill	102366	485841.0	13.935294	356425.0	6010.0	69481.0	812.0

D. Analysis

For each of the following, determine the strength of the correlation between each pair of variables. Compute the correlation strength by calculating the Pearson correlation coefficient R for pairs of columns in your DataFrame. For example, if you have a DataFrame df with each row representing a distinct county, and columns named ‘TotalCases’ and ‘Poverty’, then you can compute R like this:

```
R = df['TotalCases'].corr(df['Poverty'])
```

For any R that is > 0.5 or < -0.5 also display a scatter plot (see pandas scatterplot and seaborn documentation for information about how to display scatter plots from DataFrame data).

The COVID numbers should be normalized to population (# of cases per 100,000 residents) so that different sized counties are comparable. So for example, “COVID total cases” below really means “((COVID total cases in county * 100000) / population of county)”.

1. Across all of the counties in the State of Oregon

a. COVID total cases vs. % population in poverty
   0.24437821395123033

b. COVID total deaths vs. % population in poverty
   0.32704290486477905

c. COVID total cases vs. Per Capita Income level
d. COVID total cases vs. Per Capita Income level
e. COVID cases during December 2020 vs. % population in poverty
f. COVID deaths during December 2020 vs. % population in poverty
g. COVID cases during December 2020 vs. Per Capita Income level
h. COVID cases during December 2020 vs. Per Capita Income level

2. Across all of the counties in the entire USA
   a. COVID total cases vs. % population in poverty
   b. COVID total deaths vs. % population in poverty
   c. COVID total cases vs. Per Capita Income level
   d. COVID total cases vs. Per Capita Income level
   e. COVID cases during December 2020 vs. % population in poverty
   f. COVID deaths during December 2020 vs. % population in poverty
   g. COVID cases during December 2020 vs. Per Capita Income level
   h. COVID cases during December 2020 vs. Per Capita Income level

Note that this exercise does not constitute a competent, thorough statistical analysis of the relationships between immunological data and demographic data. It is just an illustration of the types of computations that might be accomplished with an integrated data set.