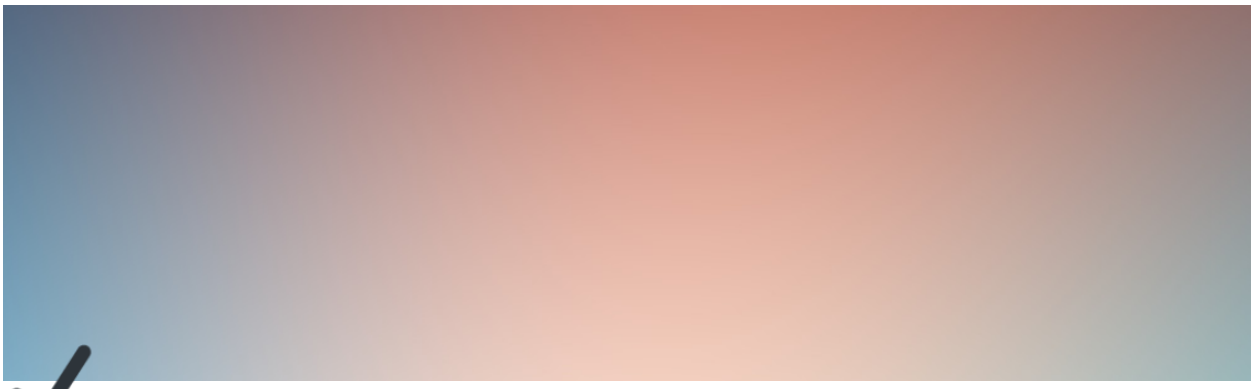

Final Exam

Contents

README INSTRUCTIONS

Run python file by typing 'python pandas_exam.py' in the command line

ALL DATA and merged json are found in the DATA folder

MAP REDUCE

(50 points) Part 1: MapReduce
Count bigrams: Take the word count from the previous problem and extend it to count bigrams. bigrams are sequences of two consecutive words. (Recall our lecture on shingles.) Don't worry about doing anything fancy in terms of tokenization. You can use Java's StringTokenizer.

1. (10 points) How many unique bigrams are there? 42035
a. Give 3 examples of such bigrams.

{('1lt', 'is'): 1}, {('a', 'truth'): 1}, {('Chapter', '1lt'): 1}

2. (5 points) List the top ten most frequent bigrams and their counts.

'of', 'the') : 462

('to', 'be') : 437

('in', 'the') : 367

('I', 'am') : 303

('Mr', 'Darcy') : 273

('of', 'her') : 261

('to', 'the') : 247

('of', 'his') : 234

('had', 'been') : 200

('I', 'have') : 188

3. (10 points) What fraction of all bigrams occurrences does the top ten bigrams account for? That is, what is the cumulative frequency of the top ten bigrams?

2972/122817

4. (5 points) How many bigrams appear only once?

42035

a. Give 3 examples of bigrams that appear only once.

{('a', 'truth'): 1}

{('Chapter', '1lt'): 1}

{('1lt', 'is'): 1}

5. (10 points) What are the five most frequent words following the word "light"? What is the frequency of observing each word?

a. Describe in detail the procedure that you used to answer this question.

6. (10 points) Running in the cloud. Otherwise, 0 points.

AWS ec2 ubuntu

```
[hadoop@ip-172-31-39-195:~$ ls
 'BigramCount$MyMapper.class'    BigramCount.class   BigramCount.java   hadoop-3.2.1          hadooptmpdata
 'BigramCount$MyReducer.class'   BigramCount.jar     book.txt           hadoop-3.2.1.tar.gz   hdfs
```

Part 2: Pandas

For this problem use the data crawled from your Project 2.

1. (5 points) Describe the that you use to solve this problem. (Hint: get it from your report.)

I built a web scraper for each website to scrape the necessary data and organize the data accordingly. All of my data is in csv format.

2. (5 points) Import the Pandas package.

```
#2. (5 points) Import the Pandas package.
import pandas as pd
```

3. (5 points) Load data into separate Data Frames. (All of you have collected multiple files.
Work with at least 2 files for this problem.)

The code below shows that the two files csv data is loaded into separate data frames. In this case or_df is a dataframe for oregons data and wa_df is the dataframe for washington's data.

a. Use csv functions to load data from csv files if your data is in csv.
b. Use json functions to load data from json documents if your data is in json.

```
#3. (5 points) Load data into separate Data Frames.
or_df = pd.read_csv("data/OR_DataProjection.csv")

#3. (5 points) Load data into separate Data Frames.
wa_df = pd.read_csv("data/WA_DataProjection.csv")
```

```
In [136]: print(wa_df)
              V1 location_name  location_id  ... icuover_upper  deaths
cases
date                                         ...
2020-01-06  28626    Washington          570  ...             0       0
0
2020-01-07  28627    Washington          570  ...             0       0
0
2020-01-08  28628    Washington          570  ...             0       0
0
2020-01-09  28629    Washington          570  ...             0       0
0
2020-01-10  28630    Washington          570  ...             0       0
0
...           ...           ...          ...  ...           ...     ...
...
2020-04-25  28736    Washington          570  ...             0   13484
743
2020-04-26  28737    Washington          570  ...             0   13663
757
2020-04-27  28738    Washington          570  ...             0   13864
771
2020-04-28  28739    Washington          570  ...             0   14059
792
2020-04-29  28740    Washington          570  ...             0   14378
805

[115 rows x 33 columns]
```

```
In [135]: print(or_df)
              V1 location_name  location_id  ... icuover_upper  deaths
cases
date                                         ...
2020-01-06  20680        Oregon          560  ...             0     0.0
0.0
2020-01-07  20681        Oregon          560  ...             0     0.0
0.0
2020-01-08  20682        Oregon          560  ...             0     0.0
0.0
2020-01-09  20683        Oregon          560  ...             0     0.0
0.0
2020-01-10  20684        Oregon          560  ...             0     0.0
0.0
...           ...           ...          ...  ...           ...     ...
...
2020-04-26  20791        Oregon          560  ...             0  2311.0
91.0
2020-04-27  20792        Oregon          560  ...             0  2354.0
92.0
2020-04-28  20793        Oregon          560  ...             0  2385.0
99.0
2020-04-29  20794        Oregon          560  ...             0  2446.0
101.0
2020-04-30  20795        Oregon          560  ...             0     NaN
NaN

[116 rows x 33 columns]
```

The images show the data frame of Washington and Oregon data files.

4. (5 points) Check the data-type of each of column by outputting the dttypes attribute of your DataFrame.

```
#4. (5 points) Check the data-type of each of column by outputting the dttypes attribute of your DataFrame.
print(or_df.dtypes)

#4. (5 points) Check the data-type of each of column by outputting the dttypes attribute of your DataFrame.
print(wa_df.dtypes)
```

```
totdea_upper       int64
bedover_mean       int64
bedover_lower      int64
bedover_upper      int64
icuover_mean       int64
icuover_lower      int64
icuover_upper      int64
deaths           float64
cases            float64
dtype: object
```

```
V1                  int64
location_name      object
location_id         int64
date               object
allbed_mean       float64
allbed_lower      float64
allbed_upper      float64
ICUbed_mean       float64
ICUbed_lower      float64
ICUbed_upper      float64
InvVen_mean       float64
InvVen_lower      float64
InvVen_upper      float64
deaths_mean       float64
deaths_lower        int64
deaths_upper      float64
admis_mean        float64
admis_lower       float64
admis_upper       float64
newICU_mean       float64
newICU_lower      float64
newICU_upper      float64
totdea_mean       float64
totdea_lower        int64
totdea_upper      float64
bedover_mean        int64
bedover_lower       int64
bedover_upper       int64
icuover_mean        int64
icuover_lower       int64
icuover_upper       int64
deaths              int64
cases               int64
dtype: object
```

5. (5 points) Show an example of sorting one of your DataFrames by a column. Give the top-15 entries in descending order.

The code below shows 15 entries in descending order

```
#5. (5 points) Show an example of sorting one of your DataFrames by a column. Give thetop-15 entries in descending order.
or_df_sorted_descending = or_df.sort_values(['cases'], ascending=False)
```

```
print(or_df_sorted_descending.head(15))
```

```
dtype: object
        V1 location_name  location_id  ... icuover_upper  deaths  cases
114  20794        Oregon          560  ...             0  2446.0  101.0
113  20793        Oregon          560  ...             0  2385.0   99.0
112  20792        Oregon          560  ...             0  2354.0   92.0
111  20791        Oregon          560  ...             0  2311.0   91.0
110  20790        Oregon          560  ...             0  2253.0   87.0
109  20789        Oregon          560  ...             0  2177.0   86.0
108  20788        Oregon          560  ...             0  2127.0   83.0
107  20787        Oregon          560  ...             0  2059.0   78.0
106  20786        Oregon          560  ...             0  2002.0   78.0
105  20785        Oregon          560  ...             0  1956.0   75.0
104  20784        Oregon          560  ...             0  1910.0   74.0
103  20783        Oregon          560  ...             0  1844.0   72.0
102  20782        Oregon          560  ...             0  1785.0   70.0
101  20781        Oregon          560  ...             0  1736.0   64.0
100  20780        Oregon          560  ...             0  1663.0   58.0
```

```
        V1 location_name  location_id  ... icuover_upper  deaths  cases
114  28740    Washington          570  ...             0   14378    805
113  28739    Washington          570  ...             0   14059    792
112  28738    Washington          570  ...             0   13864    771
111  28737    Washington          570  ...             0   13663    757
110  28736    Washington          570  ...             0   13484    743
109  28735    Washington          570  ...             0   13120    731
108  28734    Washington          570  ...             0   12906    717
107  28733    Washington          570  ...             0   12539    696
106  28732    Washington          570  ...             0   12345    683
105  28731    Washington          570  ...             0   12191    661
104  28730    Washington          570  ...             0   11805    637
103  28729    Washington          570  ...             0   11802    629
102  28728    Washington          570  ...             0   11445    610
101  28727    Washington          570  ...             0   11152    588
100  28726    Washington          570  ...             0   10783    572
```

6. (10 point) Give an example of using filtering.
a. Give an example for horizontal filtering/slicing where you select a subset of the columns. This corresponds to a projection in a SELECT statement.
b. Give an example for vertical filtering/slicing where you select a subset of the rows in your DataFramer according to some criteria. This corresponds to WHERE in SELECT statement.

```
#DESCENDING DATAFRAME
#horizontal filtering/slicing
#washing dataframe
print(wa_df_sorted_descending.loc[:, 'date':'cases'])
```

```
In [137]: print(wa_df_sorted_descending.loc[:, 'date':'cases'])
        date  allbed_mean  allbed_lower  ...  icuover_upper  deaths  cases
114  4/29/20   351.618081    149.562829  ...              0   14378    805
113  4/28/20   412.213226    239.770460  ...              0   14059    792
112  4/27/20   442.579050    299.307960  ...              0   13864    771
111  4/26/20   469.366951    356.197829  ...              0   13663    757
110  4/25/20   500.108094    417.785658  ...              0   13484    743
..       ...          ...           ...  ...            ...     ...    ...
34    2/9/20     0.000000      0.000000  ...              0       1      0
33    2/8/20     0.000000      0.000000  ...              0       1      0
32    2/7/20     0.000000      0.000000  ...              0       1      0
31    2/6/20     0.000000      0.000000  ...              0       1      0
0     1/6/20     0.000000      0.000000  ...              0       0      0

[115 rows x 30 columns]
```

```
#DESCENDING DATAFRAME
#horizontal filtering/slicing
#washing dataframe
print(or_df_sorted_descending.loc[:, 'date':'cases'])
```

```
In [138]: print(or_df_sorted_descending.loc[:, 'date':'cases'])
        date  allbed_mean  allbed_lower  ...  icuover_upper  deaths  cases
114  4/29/20     99.95590      38.69875  ...              0  2446.0  101.0
113  4/28/20     98.75735      46.39875  ...              0  2385.0   99.0
112  4/27/20    106.65070      63.89375  ...              0  2354.0   92.0
111  4/26/20    106.32305      73.64500  ...              0  2311.0   91.0
110  4/25/20    108.41015      85.54375  ...              0  2253.0   87.0
..       ...          ...           ...  ...            ...     ...    ...
33    2/8/20      0.00000       0.00000  ...              0     0.0    0.0
32    2/7/20      0.00000       0.00000  ...              0     0.0    0.0
31    2/6/20      0.00000       0.00000  ...              0     0.0    0.0
57    3/3/20      0.00000       0.00000  ...              0     2.0    0.0
115  4/30/20     80.82120      13.64750  ...              0     NaN    NaN

[116 rows x 30 columns]
```

```
#vertical filtering/slicing for DESCENDING DATAFRAME
print(wa_df_sorted_descending.loc[0:14, :])
```

```
print(or_df_sorted_descending.loc[0:19, :])
```

```
In [143]: print(or_df_sorted_descending.loc[0:19, :])
       V1 location_name  location_id  ... icuover_upper  deaths  cases
0   20680        Oregon          560  ...             0     0.0    0.0
1   20681        Oregon          560  ...             0     0.0    0.0
28  20708        Oregon          560  ...             0     0.0    0.0
26  20706        Oregon          560  ...             0     0.0    0.0
25  20705        Oregon          560  ...             0     0.0    0.0
24  20704        Oregon          560  ...             0     0.0    0.0
23  20703        Oregon          560  ...             0     0.0    0.0
22  20702        Oregon          560  ...             0     0.0    0.0
21  20701        Oregon          560  ...             0     0.0    0.0
20  20700        Oregon          560  ...             0     0.0    0.0
19  20699        Oregon          560  ...             0     0.0    0.0

[11 rows x 33 columns]
```

```
In [139]: print(wa_df_sorted_descending.loc[0:14, :])
Empty DataFrame
Columns: [V1, location_name, location_id, date, allbed_mean, allbed_lower,
allbed_upper, ICUbed_mean, ICUbed_lower, ICUbed_upper, InvVen_mean,
InvVen_lower, InvVen_upper, deaths_mean, deaths_lower, deaths_upper,
admis_mean, admis_lower, admis_upper, newICU_mean, newICU_lower,
newICU_upper, totdea_mean, totdea_lower, totdea_upper, bedover_mean,
bedover_lower, bedover_upper, icuover_mean, icuover_lower, icuover_upper,
deaths, cases]
Index: []
```

```
#ASCENDING DATAFRAME
#horizontal filtering/slicing
print(wa_df.loc[:, 'date':'cases'])
print(or_df.loc[:, 'date':'cases'])

#vertical filtering/slicing
print(wa_df.loc[0:14, :])
print(or_df.loc[0:19, :])
```

7. (10 points) Show an example where you merge two DataFrames.

I will merge the executive order dataframe into the project dataframes. Output of the dataframe is shown below in the screenshot

```
#Oregon Executive Order
or_ex = pd.read_csv("data/OR_executive_orders.csv")
print(or_ex.head(20))
or_df.set_index(pd.to_datetime(or_df['date']), inplace=True)
or_ex.set_index(pd.to_datetime(or_ex['Date']), inplace=True)

#mergininh executive order or_ex and or_df dataframe
OR_merged_df = pd.merge(or_ex, or_df, how='outer', left_index=True, right_index=True)
OR_merged_df = OR_merged_df.drop(columns='date')
OR_merged_df = OR_merged_df.drop(columns='Date')
```

3/10/20	Governor Ins	28690	Washington	570	102.135932	92.2234868	111.897895	28.0246969	26.5841886	29.5974342	25.654867	24.4110965	26.9448684	2	2	2	18.9553854	14.7894737
3/10/20	Governor Ins	28690	Washington	570	102.135932	92.2234868	111.897895	28.0246969	26.5841886	29.5974342	25.654867	24.4110965	26.9448684	2	2	2	18.9553854	14.7894737
3/10/20	The Governo	28690	Washington	570	102.135932	92.2234868	111.897895	28.0246969	26.5841886	29.5974342	25.654867	24.4110965	26.9448684	2	2	2	18.9553854	14.7894737
3/10/20	The Washing	28690	Washington	570	102.135932	92.2234868	111.897895	28.0246969	26.5841886	29.5974342	25.654867	24.4110965	26.9448684	2	2	2	18.9553854	14.7894737
3/11/20	Governor Ins	28691	Washington	570	132.658217	121.474408	144.8325	36.5565306	35.0707895	38.2107895	34.0457747	32.7570395	35.4002632	4	4	4	44.19877	38.2655921
3/12/20	Governor Ins	28692	Washington	570	154.343542	142.52954	167.213618	40.9298715	39.3604605	42.4876316	38.3249148	36.9996711	39.6397368	4	4	4	32.7686688	27.9621711
3/13/20	The Governo	28693	Washington	570	159.690768	147.337763	172.051776	42.7689389	41.0683553	44.4475658	39.9424207	38.5313816	41.3945175	2	2	2	23.2592776	19.0094518
3/14/20	he Washingt	28694	Washington	570	189.178327	175.913421	203.141053	52.0207557	50.0385526	53.955329	48.6057968	46.9972368	50.2659211	3	3	3	41.4036788	35.5331579
3/15/20	The Governo	28695	Washington	570	235.233896	219.566908	251.444605	62.9510909	60.9205921	65.0502632	59.0273203	57.3498026	60.7371053	4	4	4	56.0997761	49.1146711
3/16/20	The Governo	28696	Washington	570	250.601122	234.264803	266.568421	65.8136523	63.6420395	68.0080263	61.6600311	59.9625	63.4526974	10	10	10	27.9856942	23.1608991
3/17/20	The	28697	Washington	570	278.912982	262.218816	295.902566	68.8641908	66.6735526	71.2263158	64.1745422	62.3288816	66.1530263	8	8	8	52.4281779	45.6302632
3/17/20	The Governo	28697	Washington	570	278.912982	262.218816	295.902566	68.8641908	66.6735526	71.2263158	64.1745422	62.3288816	66.1530263	8	8	8	52.4281779	45.6302632
3/17/20	The Governo	28697	Washington	570	278.912982	262.218816	295.902566	68.8641908	66.6735526	71.2263158	64.1745422	62.3288816	66.1530263	8	8	8	52.4281779	45.6302632
3/17/20	The Washing	28697	Washington	570	278.912982	262.218816	295.902566	68.8641908	66.6735526	71.2263158	64.1745422	62.3288816	66.1530263	8	8	8	52.4281779	45.6302632
3/18/20		28698	Washington	570	257.101742	241.194474	274.653553	68.7556255	66.2683553	71.3319737	63.0334276	61.0471053	65.2351974	5	5	5	18.1357125	14.5894079
3/19/20	The Governo	28699	Washington	570	281.266793	265.080724	298.077961	75.319821	72.7483553	77.9161184	69.4477186	67.4130263	71.5553947	9	9	9	52.0033846	45.3684211
3/19/20	The Governo	28699	Washington	570	281.266793	265.080724	298.077961	75.319821	72.7483553	77.9161184	69.4477186	67.4130263	71.5553947	9	9	9	52.0033846	45.3684211
3/19/20	he Centers fc	28699	Washington	570	281.266793	265.080724	298.077961	75.319821	72.7483553	77.9161184	69.4477186	67.4130263	71.5553947	9	9	9	52.0033846	45.3684211
3/19/20	The Governo	28699	Washington	570	281.266793	265.080724	298.077961	75.319821	72.7483553	77.9161184	69.4477186	67.4130263	71.5553947	9	9	9	52.0033846	45.3684211
3/20/20		28700	Washington	570	300.744913	283.91079	319.270592	78.8564219	76.1969079	81.6088158	72.318446	70.1653947	74.5686842	12	12	12	45.8748361	39.5576974
3/21/20	The	28701	Washington	570	331.90862	314.27125	349.936908	83.4497952	80.8076974	86.2267763	76.7531298	74.5365132	79.0028947	6	6	6	73.5168479	65.565329
3/22/20	The Governo	28702	Washington	570	397.307448	379.631118	417.061579	103.773992	101.060461	106.681711	96.4697827	94.1272149	98.8951974	11	11	11	111.607169	102.357632
3/22/20	The White H	28702	Washington	570	397.307448	379.631118	417.061579	103.773992	101.060461	106.681711	96.4697827	94.1272149	98.8951974	11	11	11	111.607169	102.357632
3/22/20	President Tru	28702	Washington	570	397.307448	379.631118	417.061579	103.773992	101.060461	106.681711	96.4697827	94.1272149	98.8951974	11	11	11	111.607169	102.357632
3/22/20	The Washing	28702	Washington	570	397.307448	379.631118	417.061579	103.773992	101.060461	106.681711	96.4697827	94.1272149	98.8951974	11	11	11	111.607169	102.357632
3/23/20	The Governo	28703	Washington	570	439.102691	420.677829	460.935132	111.017253	108.14204	114.273377	103.114601	100.602566	105.585	4	4	4	74.2167421	66.3574342
3/23/20	The Governo	28703	Washington	570	439.102691	420.677829	460.935132	111.017253	108.14204	114.273377	103.114601	100.602566	105.585	4	4	4	74.2167421	66.3574342
3/24/20	he Governor	28704	Washington	570	469.969522	449.648553	492.515	127.469044	124.207697	130.837434	118.39754	115.756908	121.202829	11	11	11	73.7584073	65.842829
3/24/20	The	28704	Washington	570	469.969522	449.648553	492.515	127.469044	124.207697	130.837434	118.39754	115.756908	121.202829	11	11	11	73.7584073	65.842829

```
#Washington Executive Order data frame
wa_ex = pd.read_csv("data/WA_executive_orders.csv")
print(wa_ex.head(20))

wa_df.set_index(pd.to_datetime(wa_df['date']), inplace=True)
wa_ex.set_index(pd.to_datetime(wa_ex['Date']), inplace=True)

#mergininh executive order wa_ex and wa_df dataframe
WA_merged_df = pd.merge(wa_ex, wa_df, how='outer', left_index=True, right_index=True)
WA_merged_df = WA_merged_df.drop(columns='date')
WA_merged_df = WA_merged_df.drop(columns='Date')
WA_merged_df.columns.name = 'Date'
```


3/10/20	Governor Ins	28690	Washington	570	102.135932	92.2234868	111.897895	28.0246969	26.5841886	29.5974342	25.654867	24.4110965	26.9448684	2	2	2	18.9553854	14.7894737
3/10/20	Governor Ins	28690	Washington	570	102.135932	92.2234868	111.897895	28.0246969	26.5841886	29.5974342	25.654867	24.4110965	26.9448684	2	2	2	18.9553854	14.7894737
3/10/20	The Governo	28690	Washington	570	102.135932	92.2234868	111.897895	28.0246969	26.5841886	29.5974342	25.654867	24.4110965	26.9448684	2	2	2	18.9553854	14.7894737
3/10/20	The Washing	28690	Washington	570	102.135932	92.2234868	111.897895	28.0246969	26.5841886	29.5974342	25.654867	24.4110965	26.9448684	2	2	2	18.9553854	14.7894737
3/11/20	Governor Ins	28691	Washington	570	132.658217	121.474408	144.8325	36.5565306	35.0707895	38.2107895	34.0457747	32.7570395	35.4002632	4	4	4	44.19877	38.2655921
3/12/20	Governor Ins	28692	Washington	570	154.343542	142.52954	167.213618	40.9298715	39.3604605	42.4876316	38.3249148	36.9996711	39.6397368	4	4	4	32.7686688	27.9621711
3/13/20	The Governo	28693	Washington	570	159.690768	147.337763	172.051776	42.7689389	41.0683553	44.4475658	39.9424207	38.5313816	41.3945175	2	2	2	23.2592776	19.0094518
3/14/20	he Washingt	28694	Washington	570	189.178327	175.913421	203.141053	52.0207557	50.0385526	53.955329	48.6057968	46.9972368	50.2659211	3	3	3	41.4036788	35.5331579
3/15/20	The Governo	28695	Washington	570	235.233896	219.566908	251.444605	62.9510909	60.9205921	65.0502632	59.0273203	57.3498026	60.7371053	4	4	4	56.0997761	49.1146711
3/16/20	The Governo	28696	Washington	570	250.601122	234.264803	266.568421	65.8136523	63.6420395	68.0080263	61.6600311	59.9625	63.4526974	10	10	10	27.9856942	23.1608991
3/17/20	The	28697	Washington	570	278.912982	262.218816	295.902566	68.8641908	66.6735526	71.2263158	64.1745422	62.3288816	66.1530263	8	8	8	52.4281779	45.6302632
3/17/20	The Governo	28697	Washington	570	278.912982	262.218816	295.902566	68.8641908	66.6735526	71.2263158	64.1745422	62.3288816	66.1530263	8	8	8	52.4281779	45.6302632
3/17/20	The Governo	28697	Washington	570	278.912982	262.218816	295.902566	68.8641908	66.6735526	71.2263158	64.1745422	62.3288816	66.1530263	8	8	8	52.4281779	45.6302632
3/17/20	The Washing	28697	Washington	570	278.912982	262.218816	295.902566	68.8641908	66.6735526	71.2263158	64.1745422	62.3288816	66.1530263	8	8	8	52.4281779	45.6302632
3/18/20		28698	Washington	570	257.101742	241.194474	274.653553	68.7556255	66.2683553	71.3319737	63.0334276	61.0471053	65.2351974	5	5	5	18.1357125	14.5894079
3/19/20	The Governo	28699	Washington	570	281.266793	265.080724	298.077961	75.319821	72.7483553	77.9161184	69.4477186	67.4130263	71.5553947	9	9	9	52.0033846	45.3684211
3/19/20	The Governo	28699	Washington	570	281.266793	265.080724	298.077961	75.319821	72.7483553	77.9161184	69.4477186	67.4130263	71.5553947	9	9	9	52.0033846	45.3684211
3/19/20	he Centers fc	28699	Washington	570	281.266793	265.080724	298.077961	75.319821	72.7483553	77.9161184	69.4477186	67.4130263	71.5553947	9	9	9	52.0033846	45.3684211
3/19/20	The Governo	28699	Washington	570	281.266793	265.080724	298.077961	75.319821	72.7483553	77.9161184	69.4477186	67.4130263	71.5553947	9	9	9	52.0033846	45.3684211
3/20/20		28700	Washington	570	300.744913	283.91079	319.270592	78.8564219	76.1969079	81.6088158	72.318446	70.1653947	74.5686842	12	12	12	45.8748361	39.5576974
3/21/20	The	28701	Washington	570	331.90862	314.27125	349.936908	83.4497952	80.8076974	86.2267763	76.7531298	74.5365132	79.0028947	6	6	6	73.5168479	65.565329
3/22/20	The Governo	28702	Washington	570	397.307448	379.631118	417.061579	103.773992	101.060461	106.681711	96.4697827	94.1272149	98.8951974	11	11	11	111.607169	102.357632
3/22/20	The White H	28702	Washington	570	397.307448	379.631118	417.061579	103.773992	101.060461	106.681711	96.4697827	94.1272149	98.8951974	11	11	11	111.607169	102.357632
3/22/20	President Tru	28702	Washington	570	397.307448	379.631118	417.061579	103.773992	101.060461	106.681711	96.4697827	94.1272149	98.8951974	11	11	11	111.607169	102.357632
3/22/20	The Washing	28702	Washington	570	397.307448	379.631118	417.061579	103.773992	101.060461	106.681711	96.4697827	94.1272149	98.8951974	11	11	11	111.607169	102.357632
3/23/20	The Governo	28703	Washington	570	439.102691	420.677829	460.935132	111.017253	108.14204	114.273377	103.114601	100.602566	105.585	4	4	4	74.2167421	66.3574342
3/23/20	The Governo	28703	Washington	570	439.102691	420.677829	460.935132	111.017253	108.14204	114.273377	103.114601	100.602566	105.585	4	4	4	74.2167421	66.3574342
3/24/20	he Governor	28704	Washington	570	469.969522	449.648553	492.515	127.469044	124.207697	130.837434	118.39754	115.756908	121.202829	11	11	11	73.7584073	65.842829
3/24/20	The	28704	Washington	570	469.969522	449.648553	492.515	127.469044	124.207697	130.837434	118.39754	115.756908	121.202829	11	11	11	73.7584073	65.842829

8. (5 points) Export the merged DataFrame to

a. A csv file if your input data is in json documents.

b. A json file if your input data is in csv files.

```
#8 Export the merged DataFrame to JSON
```

```
WA_merged_df.to_json('data/WA_merged_json_records.json', orient = 'records')
```