

▼ written material

going to grab this data from gh: https://raw.githubusercontent.com/stefanbund/py3100/main/ProductList_118.csv

This python creates interactive and visually appealing geographical visualization like choropleth.

```
!pip install plotly-geo
```

```
Requirement already satisfied: plotly-geo in /usr/local/lib/python3.10/dist-packages (1.0.0)
```

Pandas and numpy are popular Python libraries. Pandas is used for data manipulation and people who work in the Data science industry. Numpy creates dataframe and helps with calculating the arrays and the data.

```
import pandas as pd
import numpy as np
```

These two url's below contain th raw data from Github that we are going to use for this project.

```
url = 'https://raw.githubusercontent.com/stefanbund/py3100/main/ProductList_118.csv'
url_m = 'https://raw.githubusercontent.com/stefanbund/py3100/main/matrix.csv'
```

This code reads the csv file and creates a Pandas dataframe so we can analyze the data.

```
df_m = pd.read_csv(url_m) #make a pandas dataframe
```

This python displays the raw data from Github into an organized dataframe.

```
df_m
```

| | City | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ... | 32 | 33 | |
|---|------------|------|------|------|------|------|------|------|------|------|-----|------|------|---|
| 0 | Birmingham | 8285 | 5343 | 6738 | 6635 | 5658 | 8118 | 4311 | 8535 | 3436 | ... | 1340 | 6923 | 3 |
| 1 | Montgomery | 1287 | 6585 | 8300 | 8874 | 8208 | 5363 | 3552 | 3387 | 2765 | ... | 4424 | 8813 | 6 |
| 2 | Mobile | 8035 | 5569 | 9492 | 5905 | 5024 | 1107 | 6937 | 5580 | 8044 | ... | 5430 | 1601 | 9 |
| 3 | Huntsville | 6280 | 2841 | 3399 | 5448 | 6173 | 5451 | 7488 | 9981 | 5236 | ... | 9169 | 7829 | 6 |
| 4 | Tuscaloosa | 4079 | 1066 | 3923 | 4177 | 4277 | 4219 | 9436 | 8160 | 4302 | ... | 1556 | 5533 | 1 |
| 5 | Hoover | 9741 | 7377 | 9410 | 9790 | 8864 | 2522 | 5347 | 9145 | 8402 | ... | 6031 | 7673 | 8 |
| 6 | Dothan | 7646 | 2060 | 4911 | 4976 | 7851 | 4277 | 7423 | 6183 | 6641 | ... | 8253 | 1565 | 6 |
| 7 | Auburn | 4326 | 2659 | 6928 | 4656 | 1828 | 5199 | 5331 | 6294 | 3076 | ... | 6128 | 3737 | 7 |
| 8 | Decatur | 3786 | 2891 | 8124 | 2469 | 3704 | 3623 | 2409 | 8287 | 2032 | ... | 6622 | 9742 | 9 |

These are the columns so they represent the dimensionality of the matrix.

```
10 Florence 8017 3187 1128 4706 9962 7547 4440 4530 9569 ... 8306 1392 1
```

df_m.columns #dimensionality of the matrix

```
Index(['City', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12',
      '13', '14', '15', '16', '17', '18', '19', '20', '21', '22', '23', '24',
      '25', '26', '27', '28', '29', '30', '31', '32', '33', '34', '35', '36',
      '37', '38', '39', '40', '41'],
      dtype='object')
```

These are all the cities in the dataframe listed.

df_m['City'] #explore a Series inside the dataframe

```
0 Birmingham
1 Montgomery
2 Mobile
3 Huntsville
4 Tuscaloosa
5 Hoover
6 Dothan
7 Auburn
8 Decatur
9 Madison
10 Florence
11 Gadsden
12 Vestavia Hills
13 Prattville
14 Phenix City
15 Alabaster
16 Bessemer
17 Enterprise
18 Opelika
19 Homewood
20 Northport
21 Pelham
22 Trussville
23 Mountain Brook
24 Fairhope
Name: City, dtype: object
```

These are the data types which contain combinations of characters.

```
df_m.dtypes
#df_m.columns
```

```
City    object
1       int64
2       int64
3       int64
4       int64
5       int64
```

```

6      int64
7      int64
8      int64
9      int64
10     int64
11     int64
12     int64
13     int64
14     int64
15     int64
16     int64
17     int64
18     int64
19     int64
20     int64
21     int64
22     int64
23     int64
24     int64
25     int64
26     int64
27     int64
28     int64
29     int64
30     int64
31     int64
32     int64
33     int64
34     int64
35     int64
36     int64
37     int64
38     int64
39     int64
40     int64
41     int64
dtype: object

```

This python creates quantiles at 0.25, 0.5, and 0.75. The columns represent all the cities in the dataframe. The rows represent the quantiles.

```

df_3 = df_m.quantile([0.25, 0.5, 0.75], numeric_only=True, axis=1)
df_3

```

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ... | |
|-------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|-----|----|
| 0.25 | 3082.0 | 3633.0 | 2236.0 | 3473.0 | 3657.0 | 4628.0 | 4254.0 | 3588.0 | 3704.0 | 3451.0 | ... | 34 |
| 0.50 | 5343.0 | 5431.0 | 5311.0 | 5771.0 | 5131.0 | 7588.0 | 5156.0 | 5331.0 | 6589.0 | 5875.0 | ... | 64 |
| 0.75 | 7242.0 | 8074.0 | 7508.0 | 7935.0 | 7490.0 | 9145.0 | 6840.0 | 7606.0 | 8221.0 | 7783.0 | ... | 74 |

3 rows x 25 columns

T stands for transpose and it's job is to summarize many columns into one value.

```

l = df_3.T.columns #transpose, T
l

Float64Index([0.25, 0.5, 0.75], dtype='float64')

```

This code shows the 25th, 50th, and 75th quartile.

```

df_3.T.mean()

0.25    3535.24
0.50    5826.36
0.75    7953.00
dtype: float64

```

This shows the 25th quartile.

```
df_3.T[0.25].mean()

3535.24
```

This shows the 50th quartile.

```
df_3.T[0.5].mean()

5826.36
```

This shows the 75th quartile.

```
df_3.T[0.75].mean()

7953.0
```

kk shows the mean of each of the quartiles.

```
kk = df_3.T.mean()
kk #series

0.25    3535.24
0.50    5826.36
0.75    7953.00
dtype: float64
```

This code shows how much percentage of display of a store is at or below the 25th quartile, per store.

```
# n =
((df_m.iloc[:, 1:] <= kk[0.25]).sum(axis=1) / df_m.shape[1]) * 100
# print(round(n))

0    28.571429
1    21.428571
2    38.095238
3    26.190476
4    21.428571
5    16.666667
6    19.047619
7    23.809524
8    21.428571
9    28.571429
10   26.190476
11   19.047619
12   26.190476
13   23.809524
14   28.571429
15   28.571429
16   14.285714
17   19.047619
18   28.571429
19   19.047619
20   28.571429
21   23.809524
22   33.333333
23   19.047619
24   33.333333
dtype: float64
```

This code assigns 25th, 50th, and 75th quartiles to variables la, ll, and lll.

```

la = df_m['25qt'] = round(((df_m.iloc[:, 1:] <= kk[0.25]).sum(axis=1) / df_m.shape[1]) * 100,1)
ll = df_m['50qt'] = round(((df_m.iloc[:, 1:] <= kk[0.50]).sum(axis=1) / df_m.shape[1]) * 100,1)
lll = df_m['75qt'] = round(((df_m.iloc[:, 1:] <= kk[0.75]).sum(axis=1) / df_m.shape[1]) * 100,1)
print(la, ll, lll)

18    28.6
19    19.0
20    28.6
21    23.8
22    33.3
23    19.0
24    33.3
dtype: float64 0      55.8
1      55.8
2      60.5
3      51.2
4      60.5
5      34.9
6      55.8
7      51.2
8      46.5
9      48.8
10     48.8
11     41.9
12     53.5
13     44.2
14     48.8
15     41.9
16     46.5
17     41.9
18     55.8
19     41.9
20     53.5
21     51.2
22     48.8
23     53.5
24     67.4
dtype: float64 0      77.3
1      70.5
2      79.5
3      77.3
4      79.5
5      59.1
6      90.9
7      79.5
8      70.5
9      75.0
10     63.6
11     68.2
12     70.5
13     75.0
14     75.0
15     84.1
16     70.5
17     72.7
18     72.7
19     68.2
20     75.0
21     72.7
22     75.0
23     70.5
24     86.4
dtype: float64

```

This code creates a data frame.

```
# df_m
```

This code creates a table that lists each of the city with its respective quartiles.

```
end_set = ['City', '25qt', '50qt', '75qt']
df_m[end_set]
```

| | City | 25qt | 50qt | 75qt |  |
|----|----------------|------|------|------|---|
| 0 | Birmingham | 28.6 | 55.8 | 77.3 |  |
| 1 | Montgomery | 21.4 | 55.8 | 70.5 | |
| 2 | Mobile | 38.1 | 60.5 | 79.5 | |
| 3 | Huntsville | 26.2 | 51.2 | 77.3 | |
| 4 | Tuscaloosa | 21.4 | 60.5 | 79.5 | |
| 5 | Hoover | 16.7 | 34.9 | 59.1 | |
| 6 | Dothan | 19.0 | 55.8 | 90.9 | |
| 7 | Auburn | 23.8 | 51.2 | 79.5 | |
| 8 | Decatur | 21.4 | 46.5 | 70.5 | |
| 9 | Madison | 28.6 | 48.8 | 75.0 | |
| 10 | Florence | 26.2 | 48.8 | 63.6 | |
| 11 | Gadsden | 19.0 | 41.9 | 68.2 | |
| 12 | Vestavia Hills | 26.2 | 53.5 | 70.5 | |
| 13 | Prattville | 23.8 | 44.2 | 75.0 | |
| 14 | Phenix City | 28.6 | 48.8 | 75.0 | |
| 15 | Alabaster | 28.6 | 41.9 | 84.1 | |
| 16 | Bessemer | 14.3 | 46.5 | 70.5 | |
| 17 | Enterprise | 19.0 | 41.9 | 72.7 | |
| 18 | Opelika | 28.6 | 55.8 | 72.7 | |
| 19 | Homewood | 19.0 | 41.9 | 68.2 | |
| 20 | Northport | 28.6 | 53.5 | 75.0 | |
| 21 | Pelham | 23.8 | 51.2 | 72.7 | |
| 22 | Trussville | 33.3 | 48.8 | 75.0 | |
| 23 | Mountain Brook | 19.0 | 53.5 | 70.5 | |
| 24 | Fairhope | 33.3 | 67.4 | 86.4 | |

This code below creates choropleth. The data contains the zip codes that is respective to each cities.

```
#choropleth:
import pandas as pd

# Create a sample dataframe
data = {'City': ['Birmingham', 'Montgomery', 'Mobile', 'Huntsville', 'Tuscaloosa', 'Hoover', 'Dothan', 'Auburn', 'Decatur', 'Madison', 'Florence', 'Gadsden', 'Vestavia Hills', 'Prattville', 'Phenix City', 'Alabaster', 'Bessemer', 'Enterprise', 'Opelika', 'Homewood', 'Northport', 'Pelham', 'Trussville', 'Mountain Brook', 'Fairhope'],
        'Zip Code': ['35201', '36101', '36601', '35801', '35401', '35216', '36301', '36830', '35601', '35756', '35630', '35901', '35216', '36066', '36867', '35007', '35020', '36330', '36801', '35209', '35473', '35124', '35173', '35213', '36532']}

df = pd.DataFrame(data)

# Create a list of zip codes
zip_codes = ['35201', '36101', '36601', '35801', '35401', '35216', '36301', '36830', '35601', '35756', '35630', '35901', '35216', '36066', '36867', '35007', '35020', '36330', '36801', '35209', '35473', '35124', '35173', '35213', '36532']

# Add the list of zip codes as a new column to the dataframe
# df = df.assign(Zip_Codes=zip_codes)
df_m = df_m.assign(zip=zip_codes)

print(df_m)
```

| | City | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ... | \ |
|----|----------------|------|------|------|------|------|------|------|------|------|-----|---|
| 0 | Birmingham | 8285 | 5343 | 6738 | 6635 | 5658 | 8118 | 4311 | 8535 | 3436 | ... | |
| 1 | Montgomery | 1287 | 6585 | 8300 | 8874 | 8208 | 5363 | 3552 | 3387 | 2765 | ... | |
| 2 | Mobile | 8035 | 5569 | 9492 | 5905 | 5024 | 1107 | 6937 | 5580 | 8044 | ... | |
| 3 | Huntsville | 6280 | 2841 | 3399 | 5448 | 6173 | 5451 | 7488 | 9981 | 5236 | ... | |
| 4 | Tuscaloosa | 4079 | 1066 | 3923 | 4177 | 4277 | 4219 | 9436 | 8160 | 4302 | ... | |
| 5 | Hoover | 9741 | 7377 | 9410 | 9790 | 8864 | 2522 | 5347 | 9145 | 8402 | ... | |
| 6 | Dothan | 7646 | 2060 | 4911 | 4976 | 7851 | 4277 | 7423 | 6183 | 6641 | ... | |
| 7 | Auburn | 4326 | 2659 | 6928 | 4656 | 1828 | 5199 | 5331 | 6294 | 3076 | ... | |
| 8 | Decatur | 3786 | 2891 | 8124 | 2469 | 3704 | 3623 | 2409 | 8287 | 2032 | ... | |
| 9 | Madison | 1934 | 3628 | 9190 | 3275 | 9344 | 5778 | 1256 | 3523 | 1781 | ... | |
| 10 | Florence | 8017 | 3187 | 1128 | 4706 | 9962 | 7547 | 4440 | 4530 | 9569 | ... | |
| 11 | Gadsden | 2290 | 6402 | 8598 | 7547 | 5158 | 9731 | 8038 | 4435 | 7357 | ... | |
| 12 | Vestavia Hills | 9471 | 9142 | 4419 | 3846 | 2016 | 5069 | 4853 | 6336 | 9062 | ... | |
| 13 | Prattville | 6039 | 8003 | 6180 | 4610 | 3548 | 7115 | 6720 | 8512 | 9954 | ... | |
| 14 | Phenix City | 8788 | 8269 | 6838 | 2863 | 6753 | 6608 | 4048 | 8774 | 4513 | ... | |
| 15 | Alabaster | 1733 | 9767 | 3274 | 7125 | 7437 | 5748 | 5399 | 6513 | 3038 | ... | |
| 16 | Bessemer | 6559 | 2453 | 1578 | 5158 | 3058 | 8075 | 7066 | 8530 | 8346 | ... | |
| 17 | Enterprise | 8436 | 7800 | 7234 | 5063 | 4274 | 1948 | 7887 | 6647 | 1320 | ... | |
| 18 | Opelika | 9998 | 8953 | 7923 | 6176 | 4369 | 9503 | 2126 | 1816 | 9224 | ... | |
| 19 | Homewood | 2373 | 7188 | 9880 | 9236 | 5969 | 9998 | 8703 | 8440 | 4643 | ... | |
| 20 | Northport | 3536 | 9231 | 8651 | 6374 | 4842 | 5704 | 8484 | 6322 | 2012 | ... | |
| 21 | Pelham | 6830 | 3736 | 2734 | 6443 | 8494 | 6206 | 7290 | 8518 | 6176 | ... | |
| 22 | Trussville | 2794 | 8273 | 9174 | 2850 | 8351 | 3978 | 5995 | 4632 | 7693 | ... | |
| 23 | Mountain Brook | 8433 | 9368 | 2141 | 2357 | 6566 | 1482 | 4787 | 3900 | 6615 | ... | |
| 24 | Fairhope | 8114 | 1464 | 2811 | 3090 | 4686 | 7995 | 7676 | 1304 | 7332 | ... | |

| | 36 | 37 | 38 | 39 | 40 | 41 | 25qt | 50qt | 75qt | zip |
|----|------|------|------|------|------|------|------|------|------|-------|
| 0 | 3555 | 1341 | 1756 | 7598 | 1509 | 1861 | 28.6 | 55.8 | 77.3 | 35201 |
| 1 | 2805 | 4601 | 4449 | 5727 | 2315 | 8822 | 21.4 | 55.8 | 70.5 | 36101 |
| 2 | 9807 | 2652 | 9296 | 2815 | 4886 | 7458 | 38.1 | 60.5 | 79.5 | 36601 |
| 3 | 7935 | 2605 | 9982 | 3338 | 9116 | 3875 | 26.2 | 51.2 | 77.3 | 35801 |
| 4 | 3657 | 2158 | 4469 | 2513 | 8135 | 6963 | 21.4 | 60.5 | 79.5 | 35401 |
| 5 | 9748 | 7224 | 4628 | 8107 | 6143 | 1671 | 16.7 | 34.9 | 59.1 | 35216 |
| 6 | 5650 | 4400 | 7842 | 4006 | 9335 | 3571 | 19.0 | 55.8 | 90.9 | 36301 |
| 7 | 4387 | 6890 | 2833 | 5083 | 9707 | 2116 | 23.8 | 51.2 | 79.5 | 36830 |
| 8 | 9305 | 6509 | 6848 | 5408 | 3707 | 8744 | 21.4 | 46.5 | 70.5 | 35601 |
| 9 | 1746 | 4470 | 7054 | 6573 | 3556 | 1374 | 28.6 | 48.8 | 75.0 | 35756 |
| 10 | 5929 | 1123 | 7306 | 8746 | 4000 | 6943 | 26.2 | 48.8 | 63.6 | 35630 |
| 11 | 2549 | 5175 | 5997 | 9608 | 7230 | 9731 | 19.0 | 41.9 | 68.2 | 35901 |
| 12 | 5142 | 9619 | 9601 | 8099 | 1391 | 6276 | 26.2 | 53.5 | 70.5 | 35216 |
| 13 | 1591 | 4401 | 3457 | 4245 | 4341 | 2573 | 23.8 | 44.2 | 75.0 | 36066 |
| 14 | 3520 | 7654 | 6845 | 7738 | 3828 | 1202 | 28.6 | 48.8 | 75.0 | 36867 |
| 15 | 2479 | 9673 | 7478 | 7207 | 7006 | 3523 | 28.6 | 41.9 | 84.1 | 35007 |
| 16 | 4810 | 7641 | 5365 | 3545 | 6812 | 9483 | 14.3 | 46.5 | 70.5 | 35020 |
| 17 | 3461 | 2640 | 4375 | 8634 | 4917 | 2830 | 19.0 | 41.9 | 72.7 | 36330 |
| 18 | 5191 | 9304 | 2720 | 3100 | 3912 | 1548 | 28.6 | 55.8 | 72.7 | 36801 |
| 19 | 8787 | 5459 | 8389 | 5242 | 2224 | 6025 | 19.0 | 41.9 | 68.2 | 35209 |
| 20 | 6947 | 5401 | 6681 | 9018 | 1668 | 8307 | 28.6 | 53.5 | 75.0 | 35473 |
| 21 | 2777 | 4045 | 7309 | 4745 | 4284 | 2640 | 23.8 | 51.2 | 72.7 | 35124 |
| 22 | 1650 | 9470 | 6356 | 4700 | 3344 | 8743 | 33.3 | 48.8 | 75.0 | 35173 |
| 23 | 5765 | 3653 | 5198 | 9266 | 4945 | 3935 | 19.0 | 53.5 | 70.5 | 35213 |
| 24 | 3457 | 4808 | 7227 | 5482 | 6355 | 4553 | 33.3 | 67.4 | 86.4 | 36532 |

[25 rows x 46 columns]

This code inspects the column names of a DataFrame.

```
df_m.columns
```

```
Index(['City', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12',
      '13', '14', '15', '16', '17', '18', '19', '20', '21', '22', '23', '24',
      '25', '26', '27', '28', '29', '30', '31', '32', '33', '34', '35', '36',
      '37', '38', '39', '40', '41', '25qt', '50qt', '75qt', 'zip'],
      dtype='object')
```

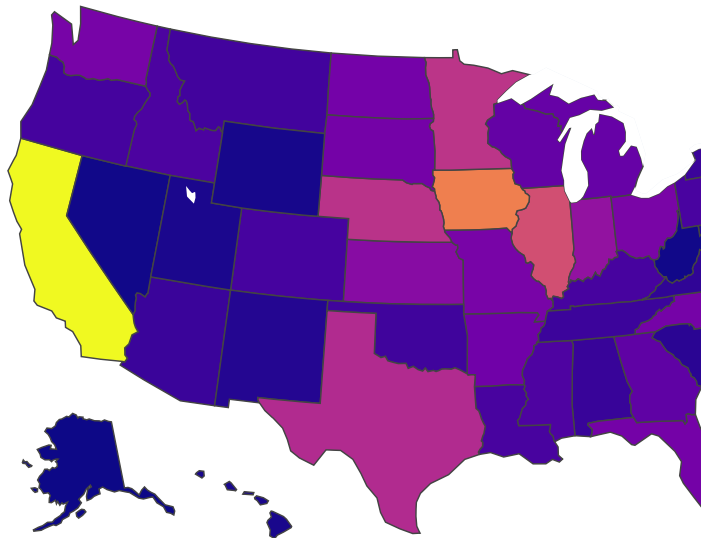
This choropleth shows all the states and its total exports. Lighter the color the higher the export. For example, California has a high number of total exports.

```
import plotly.express as px
import pandas as pd

# Load data
df_demo = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/2011_us_ag_exports.csv')

# Create choropleth map
fig = px.choropleth(df_demo, locations='code', locationmode='USA-states', color='total exports', scope='usa')

# Show map
fig.show()
```



This shows all the states and total exports. It further breaks it down to what different kind of exports.

df_demo

| | code | state | category | total exports | beef | pork | poultry | dairy | fruits fresh | fruits proc |
|----|------|----------------|----------|---------------|-------|--------|---------|--------|--------------|-------------|
| 0 | AL | Alabama | state | 1390.63 | 34.4 | 10.6 | 481.0 | 4.06 | 8.0 | |
| 1 | AK | Alaska | state | 13.31 | 0.2 | 0.1 | 0.0 | 0.19 | 0.0 | |
| 2 | AZ | Arizona | state | 1463.17 | 71.3 | 17.9 | 0.0 | 105.48 | 19.3 | |
| 3 | AR | Arkansas | state | 3586.02 | 53.2 | 29.4 | 562.9 | 3.53 | 2.2 | |
| 4 | CA | California | state | 16472.88 | 228.7 | 11.1 | 225.4 | 929.95 | 2791.8 | 59.6 |
| 5 | CO | Colorado | state | 1851.33 | 261.4 | 66.0 | 14.0 | 71.94 | 5.7 | |
| 6 | CT | Connecticut | state | 259.62 | 1.1 | 0.1 | 6.9 | 9.49 | 4.2 | |
| 7 | DE | Delaware | state | 282.19 | 0.4 | 0.6 | 114.7 | 2.30 | 0.5 | |
| 8 | FL | Florida | state | 3764.09 | 42.6 | 0.9 | 56.9 | 66.31 | 438.2 | 9.0 |
| 9 | GA | Georgia | state | 2860.84 | 31.0 | 18.9 | 630.4 | 38.38 | 74.6 | 1.0 |
| 10 | HI | Hawaii | state | 401.84 | 4.0 | 0.7 | 1.3 | 1.16 | 17.7 | |
| 11 | ID | Idaho | state | 2078.89 | 119.8 | 0.0 | 2.4 | 294.60 | 6.9 | |
| 12 | IL | Illinois | state | 8709.48 | 53.7 | 394.0 | 14.0 | 45.82 | 4.0 | |
| 13 | IN | Indiana | state | 5050.23 | 21.9 | 341.9 | 165.6 | 89.70 | 4.1 | |
| 14 | IA | Iowa | state | 11273.76 | 289.8 | 1895.6 | 155.6 | 107.00 | 1.0 | |
| 15 | KS | Kansas | state | 4589.01 | 659.3 | 179.4 | 6.4 | 65.45 | 1.0 | |
| 16 | KY | Kentucky | state | 1889.15 | 54.8 | 34.2 | 151.3 | 28.27 | 2.1 | |
| 17 | LA | Louisiana | state | 1914.23 | 19.8 | 0.8 | 77.2 | 6.02 | 5.7 | |
| 18 | ME | Maine | state | 278.37 | 1.4 | 0.5 | 10.4 | 16.18 | 16.6 | |
| 19 | MD | Maryland | state | 692.75 | 5.6 | 3.1 | 127.0 | 24.81 | 4.1 | |
| 20 | MA | Massachusetts | state | 248.65 | 0.6 | 0.5 | 0.6 | 5.81 | 25.8 | |
| 21 | MI | Michigan | state | 3164.16 | 37.7 | 118.1 | 32.6 | 214.82 | 82.3 | 1.0 |
| 22 | MN | Minnesota | state | 7192.33 | 112.3 | 740.4 | 189.2 | 218.05 | 2.5 | |
| 23 | MS | Mississippi | state | 2170.80 | 12.8 | 30.4 | 370.8 | 5.45 | 5.4 | |
| 24 | MO | Missouri | state | 3933.42 | 137.2 | 277.3 | 196.1 | 34.26 | 4.2 | |
| 25 | MT | Montana | state | 1718.00 | 105.0 | 16.7 | 1.7 | 6.82 | 1.1 | |
| 26 | NE | Nebraska | state | 7114.13 | 762.2 | 262.5 | 31.4 | 30.07 | 0.7 | |
| 27 | NV | Nevada | state | 139.89 | 21.8 | 0.2 | 0.0 | 16.57 | 0.4 | |
| 28 | NH | New Hampshire | state | 73.06 | 0.6 | 0.2 | 0.8 | 7.46 | 2.6 | |
| 29 | NJ | New Jersey | state | 500.40 | 0.8 | 0.4 | 4.6 | 3.37 | 35.0 | |
| 30 | NM | New Mexico | state | 751.58 | 117.2 | 0.1 | 0.3 | 191.01 | 32.6 | |
| 31 | NY | New York | state | 1488.90 | 22.2 | 5.8 | 17.7 | 331.80 | 64.7 | 1.0 |
| 32 | NC | North Carolina | state | 3806.05 | 24.8 | 702.8 | 598.4 | 24.90 | 23.8 | |
| 33 | ND | North Dakota | state | 3761.96 | 78.5 | 16.1 | 0.5 | 8.14 | 0.1 | |
| 34 | OH | Ohio | state | 3979.79 | 36.2 | 199.1 | 129.9 | 134.57 | 8.7 | |
| 35 | OK | Oklahoma | state | 1646.41 | 337.6 | 265.3 | 131.1 | 24.35 | 3.0 | |

This code displays the cloumn names.

| | | | | | | | | | | |
|----|----|--------------|-------|---------|------|------|-------|--------|------|--|
| 37 | PA | Pennsylvania | state | 1969.87 | 50.9 | 91.3 | 169.8 | 280.87 | 28.6 | |
|----|----|--------------|-------|---------|------|------|-------|--------|------|--|

df_demo.columns

```
Index(['code', 'state', 'category', 'total exports', 'beef', 'pork', 'poultry',  
      'dairy', 'fruits fresh', 'fruits proc', 'total fruits', 'veggies fresh',
```

```
'veggies proc', 'total veggies', 'corn', 'wheat', 'cotton'],
dtype='object')
```

```
42 TX Texas state 6648 22 961 0 42 7 339 2 240 55 31 9 1
```

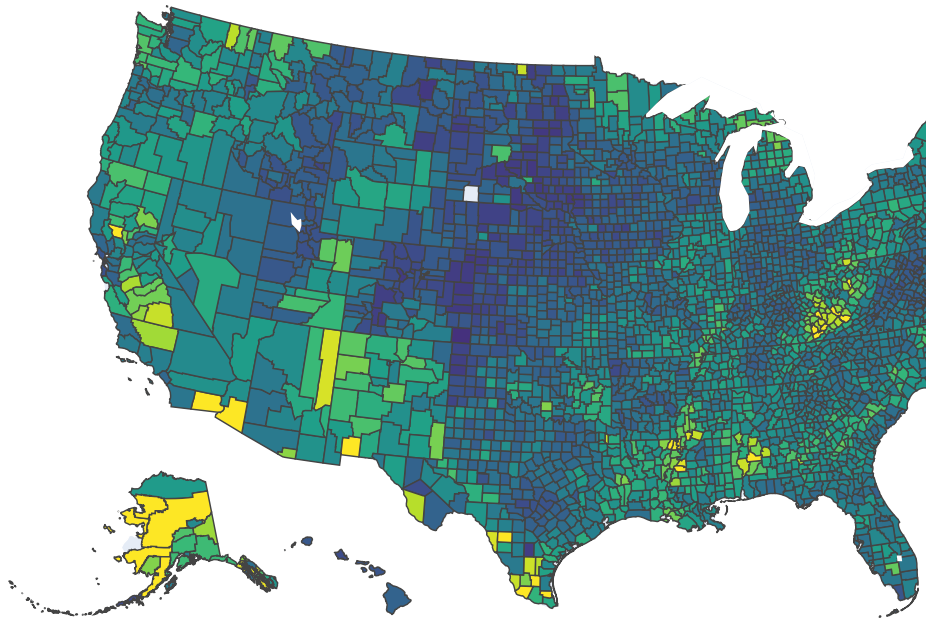
This choropleth below shows all the states with its FIPS code and unemployment number.

```
from urllib.request import urlopen
import json
with urlopen('https://raw.githubusercontent.com/plotly/datasets/master/geojson-counties-fips.json') as response:
    counties = json.load(response)

import pandas as pd
df_us = pd.read_csv("https://raw.githubusercontent.com/plotly/datasets/master/fips-unemp-16.csv",
                    dtype={"fips": str})

import plotly.express as px

fig = px.choropleth(df_us, geojson=counties, locations='fips', color='unemp',
                    color_continuous_scale="Viridis",
                    range_color=(0, 12),
                    scope="usa",
                    labels={'unemp': 'unemployment rate'})
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```



This code displays the names of the columns in the DataFrame.

```
df_us.columns

Index(['fips', 'unemp'], dtype='object')
```

This code shows the FIPS number which is the county code and the respective unemployment rate.

```
df_us
```

| | fips | unemp | |
|------|-------|-------|--|
| 0 | 01001 | 5.3 | |
| 1 | 01003 | 5.4 | |
| 2 | 01005 | 8.6 | |
| 3 | 01007 | 6.6 | |
| 4 | 01009 | 5.5 | |
| ... | ... | ... | |
| 3214 | 72145 | 13.9 | |
| 3215 | 72147 | 10.6 | |
| 3216 | 72149 | 20.2 | |
| 3217 | 72151 | 16.9 | |

This code below shows all the FIPS codes that is associated with each store in Alabama.

3219 rows x 2 columns

```
al_fips = [
    {'County': 'Autauga', 'FIPS Code': '01001'},
    {'County': 'Baldwin', 'FIPS Code': '01003'},
    {'County': 'Barbour', 'FIPS Code': '01005'},
    {'County': 'Bibb', 'FIPS Code': '01007'},
    {'County': 'Blount', 'FIPS Code': '01009'},
    {'County': 'Bullock', 'FIPS Code': '01011'},
    {'County': 'Butler', 'FIPS Code': '01013'},
    {'County': 'Calhoun', 'FIPS Code': '01015'},
    {'County': 'Chambers', 'FIPS Code': '01017'},
    {'County': 'Cherokee', 'FIPS Code': '01019'},
    {'County': 'Chilton', 'FIPS Code': '01021'},
    {'County': 'Choctaw', 'FIPS Code': '01023'},
    {'County': 'Clarke', 'FIPS Code': '01025'},
    {'County': 'Clay', 'FIPS Code': '01027'},
    {'County': 'Cleburne', 'FIPS Code': '01029'},
    {'County': 'Coffee', 'FIPS Code': '01031'},
    {'County': 'Colbert', 'FIPS Code': '01033'},
    {'County': 'Conecuh', 'FIPS Code': '01035'},
    {'County': 'Greene', 'FIPS Code': '28073'},
    {'County': 'Hale', 'FIPS Code': '28065'},
    {'County': 'Henry', 'FIPS Code': '28067'},
    {'County': 'Houston', 'FIPS Code': '28069'},
    {'County': 'Jackson', 'FIPS Code': '28071'},
    {'County': 'Jefferson', 'FIPS Code': '28073'},
    {'County': 'Lamar', 'FIPS Code': '28073'}]
len(al_fips)
```

25

This code retrieves the column names of the DataFrame.

```
df_m.columns
```

```
Index(['City', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12',
      '13', '14', '15', '16', '17', '18', '19', '20', '21', '22', '23', '24',
      '25', '26', '27', '28', '29', '30', '31', '32', '33', '34', '35', '36',
      '37', '38', '39', '40', '41', '25qt', '50qt', '75qt', 'zip'],
      dtype=object)
```

This code shows all of the displays.

```
df_m
```

| | City | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ... | 36 | 37 | |
|----|----------------|------|------|------|------|------|------|------|------|------|-----|------|------|---|
| 0 | Birmingham | 8285 | 5343 | 6738 | 6635 | 5658 | 8118 | 4311 | 8535 | 3436 | ... | 3555 | 1341 | 1 |
| 1 | Montgomery | 1287 | 6585 | 8300 | 8874 | 8208 | 5363 | 3552 | 3387 | 2765 | ... | 2805 | 4601 | 4 |
| 2 | Mobile | 8035 | 5569 | 9492 | 5905 | 5024 | 1107 | 6937 | 5580 | 8044 | ... | 9807 | 2652 | 9 |
| 3 | Huntsville | 6280 | 2841 | 3399 | 5448 | 6173 | 5451 | 7488 | 9981 | 5236 | ... | 7935 | 2605 | 9 |
| 4 | Tuscaloosa | 4079 | 1066 | 3923 | 4177 | 4277 | 4219 | 9436 | 8160 | 4302 | ... | 3657 | 2158 | 4 |
| 5 | Hoover | 9741 | 7377 | 9410 | 9790 | 8864 | 2522 | 5347 | 9145 | 8402 | ... | 9748 | 7224 | 4 |
| 6 | Dothan | 7646 | 2060 | 4911 | 4976 | 7851 | 4277 | 7423 | 6183 | 6641 | ... | 5650 | 4400 | 7 |
| 7 | Auburn | 4326 | 2659 | 6928 | 4656 | 1828 | 5199 | 5331 | 6294 | 3076 | ... | 4387 | 6890 | 2 |
| 8 | Decatur | 3786 | 2891 | 8124 | 2469 | 3704 | 3623 | 2409 | 8287 | 2032 | ... | 9305 | 6509 | 6 |
| 9 | Madison | 1934 | 3628 | 9190 | 3275 | 9344 | 5778 | 1256 | 3523 | 1781 | ... | 1746 | 4470 | 7 |
| 10 | Florence | 8017 | 3187 | 1128 | 4706 | 9962 | 7547 | 4440 | 4530 | 9569 | ... | 5929 | 1123 | 7 |
| 11 | Gadsden | 2290 | 6402 | 8598 | 7547 | 5158 | 9731 | 8038 | 4435 | 7357 | ... | 2549 | 5175 | 5 |
| 12 | Vestavia Hills | 9471 | 9142 | 4419 | 3846 | 2016 | 5069 | 4853 | 6336 | 9062 | ... | 5142 | 9619 | 9 |
| 13 | Prattville | 6039 | 8003 | 6180 | 4610 | 3548 | 7115 | 6720 | 8512 | 9954 | ... | 1591 | 4401 | 3 |
| 14 | Phenix City | 8788 | 8269 | 6838 | 2863 | 6753 | 6608 | 4048 | 8774 | 4513 | ... | 3520 | 7654 | 6 |
| 15 | Alabaster | 1733 | 9767 | 3274 | 7125 | 7437 | 5748 | 5399 | 6513 | 3038 | ... | 2479 | 9673 | 7 |
| 16 | Bessemer | 6559 | 2453 | 1578 | 5158 | 3058 | 8075 | 7066 | 8530 | 8346 | ... | 4810 | 7641 | 5 |
| 17 | Enterprise | 8436 | 7800 | 7234 | 5063 | 4274 | 1948 | 7887 | 6647 | 1320 | ... | 3461 | 2640 | 4 |
| 18 | Opelika | 9998 | 8953 | 7923 | 6176 | 4369 | 9503 | 2126 | 1816 | 9224 | ... | 5191 | 9304 | 2 |
| 19 | Homewood | 2373 | 7188 | 9880 | 9236 | 5969 | 9998 | 8703 | 8440 | 4643 | ... | 8787 | 5459 | 8 |
| 20 | Northport | 3536 | 9231 | 8651 | 6374 | 4842 | 5704 | 8484 | 6322 | 2012 | ... | 6947 | 5401 | 6 |
| 21 | Pelham | 6830 | 3736 | 2734 | 6443 | 8494 | 6206 | 7290 | 8518 | 6176 | ... | 2777 | 4045 | 7 |
| 22 | Trussville | 2794 | 8273 | 9174 | 2850 | 8351 | 3978 | 5995 | 4632 | 7693 | ... | 1650 | 9470 | 6 |
| 23 | Mountain Brook | 8433 | 9368 | 2141 | 2357 | 6566 | 1482 | 4787 | 3900 | 6615 | ... | 5765 | 3653 | 5 |
| 24 | Fairhope | 8114 | 1464 | 2811 | 3090 | 4686 | 7995 | 7676 | 1304 | 7332 | ... | 3457 | 4808 | 7 |

df_m.shape

This code will return an integer representing the total number of rows in the DataFrame df_m.

```
df_m.shape[0]
```

25

This code is inspecting the lengths of al FLIPS and calculating the total number of elements in the DataFrame.

```
print(len(al_fips))
df_counties = pd.DataFrame(al_fips)
df_counties.size
```

25

50

This code displays the column names of the cities in the DataFrame.

```
print(df_counties.columns)

Index(['County', 'FIPS Code'], dtype='object')
```

This code below merges df_m, counties, and columns.

```
merged_df = pd.concat([df_m, df_counties], axis=1)
merged_df.head()
```

| | City | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ... | 38 | 39 | |
|---|------------|------|------|------|------|------|------|------|------|------|-----|------|------|----|
| 0 | Birmingham | 8285 | 5343 | 6738 | 6635 | 5658 | 8118 | 4311 | 8535 | 3436 | ... | 1756 | 7598 | 15 |
| 1 | Montgomery | 1287 | 6585 | 8300 | 8874 | 8208 | 5363 | 3552 | 3387 | 2765 | ... | 4449 | 5727 | 23 |
| 2 | Mobile | 8035 | 5569 | 9492 | 5905 | 5024 | 1107 | 6937 | 5580 | 8044 | ... | 9296 | 2815 | 48 |
| 3 | Huntsville | 6280 | 2841 | 3399 | 5448 | 6173 | 5451 | 7488 | 9981 | 5236 | ... | 9982 | 3338 | 91 |
| 4 | Tuscaloosa | 4079 | 1066 | 3923 | 4177 | 4277 | 4219 | 9436 | 8160 | 4302 | ... | 4469 | 2513 | 81 |

5 rows x 48 columns

This code shows the list of column names in the merged_df DataFrame.

```
merged_df.columns
```

```
Index(['City', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12',
      '13', '14', '15', '16', '17', '18', '19', '20', '21', '22', '23', '24',
      '25', '26', '27', '28', '29', '30', '31', '32', '33', '34', '35', '36',
      '37', '38', '39', '40', '41', '25qt', '50qt', '75qt', 'zip', 'County',
      'FIPS Code'],
      dtype='object')
```

This choropleth map shows which city is doing well and which cities are not within Alabama.

```
import plotly.express as px

fig = px.choropleth(merged_df, geojson=counties, locations='FIPS Code', color='25qt',
                    color_continuous_scale="Viridis",
                    range_color=(0, 38),
                    scope="usa",
                    hover_name="City",
                    hover_data=["City"],
                    labels={'25qt': 'percentage displays under 25th qt'} #
                    )
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```

This choropleth map shows every county in Alabama and how it is doing.

```
import plotly.express as px
import requests
import json
import pandas as pd

# Load the geojson data for Alabama's counties
r = requests.get('https://raw.githubusercontent.com/plotly/datasets/master/geojson-counties-fips.json')
counties = json.loads(r.text)

# Filter the geojson data to only include Alabama's counties
target_states = ['01']
counties['features'] = [f for f in counties['features'] if f['properties']['STATE'] in target_states]

# Load the sample data for Alabama's counties
df = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/fips-unemp-16.csv', dtype={'fips': str})

# Create the choropleth map
fig = px.choropleth(df, geojson=counties, locations='fips', color='unemp',
                    color_continuous_scale='Viridis', range_color=(0, 12),
                    scope='usa', labels={'unemp': 'unemployment rate'})
fig.update_layout(margin={'r': 0, 't': 0, 'l': 0, 'b': 0})
fig.show()
```

