

```
In [220... import pandas as pd
import matplotlib
import numpy as np
import seaborn as sns
from scipy import stats
from scipy.io import arff
from matplotlib import cm
import matplotlib.pyplot as plt
from matplotlib.colors import LinearSegmentedColormap
from matplotlib.colors import ListedColormap
from matplotlib.patches import Patch, Circle
import matplotlib.colors as mcolors
from matplotlib import colormaps
import matplotlib.ticker as ticker
from matplotlib.ticker import FuncFormatter
from matplotlib.colors import to_hex, to_rgba, Normalize, LogNorm, TwoSlopeNorm, Boun
import matplotlib.patches as mpatches
from mpl_toolkits.mplot3d import Axes3D
from numpy.random import randn
from numpy.random import seed
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, precision_score, recall_score
from sklearn.metrics import classification_report
from sklearn.metrics import mean_squared_error
from sklearn.tree import DecisionTreeRegressor
from bokeh.plotting import figure, output_file, show
from bokeh.models import GeoJSONDataSource
import geopandas as gpd
import folium
from PIL import Image
import plotly.graph_objects as go
import plotly.graph_objs as go
import plotly.offline as py
import os
import glob
from geopandas import clip
from shapely.geometry import MultiPolygon, Polygon
import networkx as nx
import squarify
```

```
In [221... file_path = '/Users/keriwilkins/Documents/Provisional Natality - 2023 through 02-29-2
```

```
In [222... raw_data = pd.read_excel(file_path)
```

```
In [223... print(raw_data)
```

	State of Residence	State of Residence Code	Mother's Single Race 15	\
0	Alabama	1.0	White	
1	Alabama	1.0	White	
2	Alabama	1.0	White	
3	Alabama	1.0	White	
4	Alabama	1.0	White	
...	...	...	...	
34589	Wisconsin	55.0	More than one race	
34590	Wisconsin	55.0	More than one race	
34591	Wisconsin	55.0	More than one race	
34592	Wisconsin	55.0	More than one race	
34593	Wyoming	56.0	More than one race	

	Mother's Single Race 15 Code	Number of Prenatal Visits Recode	\
0	1.0	No visits	
1	1.0	No visits	
2	1.0	No visits	
3	1.0	No visits	
4	1.0	No visits	
...	...	...	
34589	15.0	19 or more visits	
34590	15.0	19 or more visits	
34591	15.0	19 or more visits	
34592	15.0	Unknown or Not Stated	
34593	15.0	11 to 12 visits	

	Number of Prenatal Visits Recode	Code	Month Prenatal Care Began	\
0	1.0	No prenatal care		
1	1.0	No prenatal care		
2	1.0	No prenatal care		
3	1.0	No prenatal care		
4	1.0	No prenatal care		
...	...	...		
34589	11.0	2nd month		
34590	11.0	2nd month		
34591	11.0	2nd month		
34592	12.0	Unknown or Not Stated		
34593	7.0	2nd month		

	Month Prenatal Care Began Code	OE Gestational Age Weekly	\
0	0.0	28 weeks	
1	0.0	31 weeks	
2	0.0	32 weeks	
3	0.0	33 weeks	
4	0.0	34 weeks	
...	...	...	
34589	2.0	37 weeks	
34590	2.0	38 weeks	
34591	2.0	39 weeks	
34592	99.0	39 weeks	
34593	2.0	39 weeks	

	OE Gestational Age Weekly Code	Births
0	28.0	10.0
1	31.0	12.0
2	32.0	10.0
3	33.0	12.0
4	34.0	32.0
...	...	...
34589	37.0	11.0
34590	38.0	11.0
34591	39.0	11.0
34592	39.0	14.0
34593	39.0	13.0

[34594 rows x 11 columns]

```
In [224... print(len(raw_data))
```

34594

```
In [225... print((raw_data).dtypes)
```

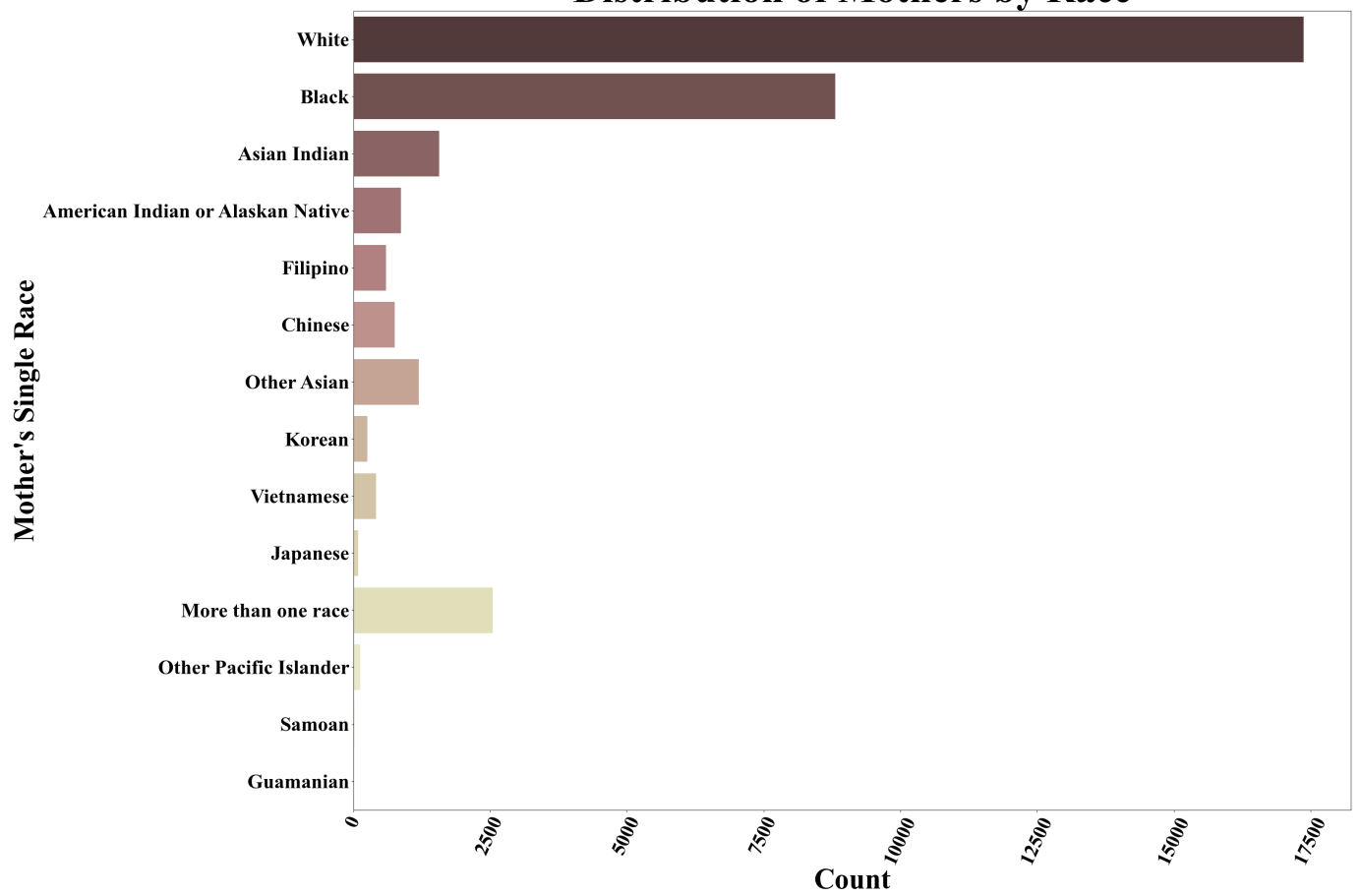
```
State of Residence          object
State of Residence Code    float64
Mother's Single Race 15    object
Mother's Single Race 15 Code float64
Number of Prenatal Visits Recode object
Number of Prenatal Visits Recode Code float64
Month Prenatal Care Began  object
Month Prenatal Care Began Code float64
OE Gestational Age Weekly  object
OE Gestational Age Weekly Code float64
Births                     float64
dtype: object
```

```
In [226... print((raw_data).columns)
```

```
Index(['State of Residence', 'State of Residence Code',
      'Mother's Single Race 15', 'Mother's Single Race 15 Code',
      'Number of Prenatal Visits Recode',
      'Number of Prenatal Visits Recode Code', 'Month Prenatal Care Began',
      'Month Prenatal Care Began Code', 'OE Gestational Age Weekly',
      'OE Gestational Age Weekly Code', 'Births'],
      dtype='object')
```

```
In [283... plt.figure(figsize=(31, 25), facecolor='none')
sns.countplot(y="Mother's Single Race 15", data=raw_data, hue="Mother's Single Race 1
plt.title('Distribution of Mothers by Race', fontsize=70, fontweight='bold', fontname
plt.xlabel('Count', fontsize=50, fontweight='bold', fontname='Times New Roman')
plt.ylabel("Mother's Single Race", fontsize=50, fontweight='bold', fontname='Times Ne
plt.xticks(rotation=65, fontsize=35, fontweight='bold', fontname='Times New Roman')
plt.yticks(fontsize=35, fontweight='bold', fontname='Times New Roman')
plt.savefig('distribution_of_mothers_by_race.png', transparent=True)
plt.show()
```

Distribution of Mothers by Race

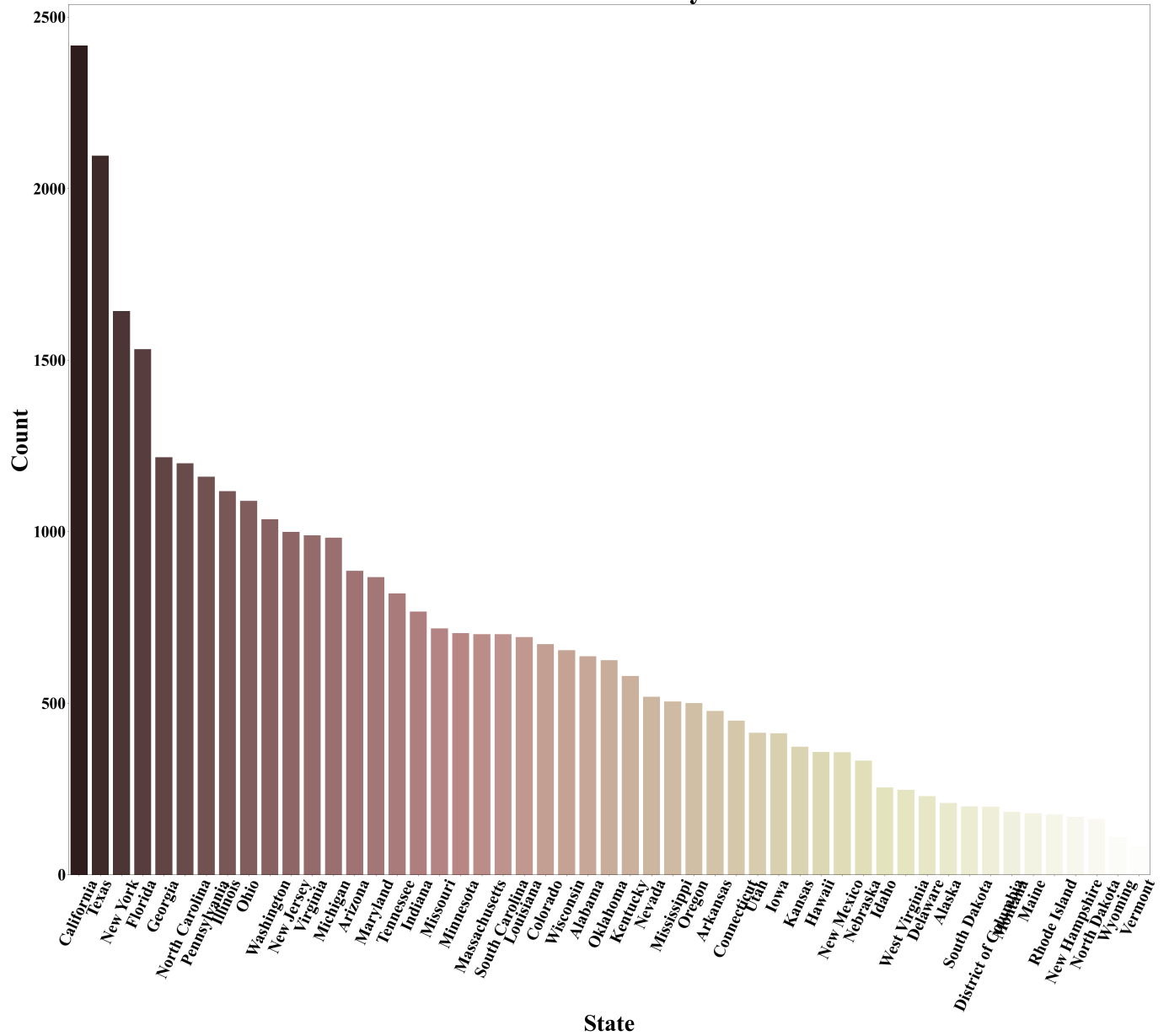


In [286...

```
plt.figure(figsize=(53, 43))
sns.barplot(x=raw_data['State of Residence'].value_counts().index,
            y=raw_data['State of Residence'].value_counts(),
            hue=raw_data['State of Residence'].value_counts().index,
            legend=False, palette='pink')
plt.title('Count of Mothers by State', fontsize=80, fontweight='bold', fontname='Time
plt.xlabel('State', fontsize=65, fontweight='bold', fontname='Times New Roman')
plt.ylabel('Count', fontsize=65, fontweight='bold', fontname='Times New Roman')
plt.xticks(rotation=65, fontsize=45, fontweight='bold', fontname='Times New Roman')
plt.yticks(fontsize=45, fontweight='bold', fontname='Times New Roman')
plt.savefig('distribution_of_mothers_by_state.png', transparent=True)

plt.show()
```

Count of Mothers by State



```
In [229...] population_data = pd.read_excel('/Users/keriwilkins/Library/CloudStorage/OneDrive-Personal/PopulationData.xlsx')

In [230...] population_data['State of Residence'] = population_data['State of Residence'].str.strip()

In [231...] print(population_data )
```

	State of Residence	2023
0	Alabama	5108468
1	Alaska	733406
2	Arizona	7431344
3	Arkansas	3067732
4	California	38965193
5	Colorado	5877610
6	Connecticut	3617176
7	Delaware	1031890
8	District of Columbia	678972
9	Florida	22610726
10	Georgia	11029227
11	Hawaii	1435138
12	Idaho	1964726
13	Illinois	12549689
14	Indiana	6862199
15	Iowa	3207004
16	Kansas	2940546
17	Kentucky	4526154
18	Louisiana	4573749
19	Maine	1395722
20	Maryland	6180253
21	Massachusetts	7001399
22	Michigan	10037261
23	Minnesota	5737915
24	Mississippi	2939690
25	Missouri	6196156
26	Montana	1132812
27	Nebraska	1978379
28	Nevada	3194176
29	New Hampshire	1402054
30	New Jersey	9290841
31	New Mexico	2114371
32	New York	19571216
33	North Carolina	10835491
34	North Dakota	783926
35	Ohio	11785935
36	Oklahoma	4053824
37	Oregon	4233358
38	Pennsylvania	12961683
39	Rhode Island	1095962
40	South Carolina	5373555
41	South Dakota	919318
42	Tennessee	7126489
43	Texas	30503301
44	Utah	3417734
45	Vermont	647464
46	Virginia	8715698
47	Washington	7812880
48	West Virginia	1770071
49	Wisconsin	5910955
50	Wyoming	584057

```
In [232... birth_counts_by_state = raw_data['State of Residence'].value_counts().reset_index()
          birth_counts_by_state.columns = ['State of Residence', 'Birth Count']
```

```
In [233... print(birth_counts_by_state)
```

	State of Residence	Birth Count
0	California	2417
1	Texas	2096
2	New York	1643
3	Florida	1532
4	Georgia	1217
5	North Carolina	1199
6	Pennsylvania	1160
7	Illinois	1118
8	Ohio	1090
9	Washington	1036
10	New Jersey	999
11	Virginia	989
12	Michigan	982
13	Arizona	886
14	Maryland	867
15	Tennessee	820
16	Indiana	767
17	Missouri	718
18	Minnesota	704
19	Massachusetts	701
20	South Carolina	701
21	Louisiana	693
22	Colorado	672
23	Wisconsin	654
24	Alabama	637
25	Oklahoma	625
26	Kentucky	579
27	Nevada	519
28	Mississippi	505
29	Oregon	500
30	Arkansas	477
31	Connecticut	449
32	Utah	414
33	Iowa	412
34	Kansas	373
35	Hawaii	358
36	New Mexico	357
37	Nebraska	332
38	Idaho	254
39	West Virginia	247
40	Delaware	229
41	Alaska	209
42	South Dakota	199
43	District of Columbia	198
44	Montana	183
45	Maine	179
46	Rhode Island	175
47	New Hampshire	168
48	North Dakota	162
49	Wyoming	110
50	Vermont	82

```
In [234... normalized_raw_data = pd.merge(population_data, birth_counts_by_state, how='left', on
```

```
In [235... # Print the merged DataFrame
print(normalized_raw_data)
```

	State of Residence	2023	Birth Count
0	Alabama	5108468	637
1	Alaska	733406	209
2	Arizona	7431344	886
3	Arkansas	3067732	477
4	California	38965193	2417
5	Colorado	5877610	672
6	Connecticut	3617176	449
7	Delaware	1031890	229
8	District of Columbia	678972	198
9	Florida	22610726	1532
10	Georgia	11029227	1217
11	Hawaii	1435138	358
12	Idaho	1964726	254
13	Illinois	12549689	1118
14	Indiana	6862199	767
15	Iowa	3207004	412
16	Kansas	2940546	373
17	Kentucky	4526154	579
18	Louisiana	4573749	693
19	Maine	1395722	179
20	Maryland	6180253	867
21	Massachusetts	7001399	701
22	Michigan	10037261	982
23	Minnesota	5737915	704
24	Mississippi	2939690	505
25	Missouri	6196156	718
26	Montana	1132812	183
27	Nebraska	1978379	332
28	Nevada	3194176	519
29	New Hampshire	1402054	168
30	New Jersey	9290841	999
31	New Mexico	2114371	357
32	New York	19571216	1643
33	North Carolina	10835491	1199
34	North Dakota	783926	162
35	Ohio	11785935	1090
36	Oklahoma	4053824	625
37	Oregon	4233358	500
38	Pennsylvania	12961683	1160
39	Rhode Island	1095962	175
40	South Carolina	5373555	701
41	South Dakota	919318	199
42	Tennessee	7126489	820
43	Texas	30503301	2096
44	Utah	3417734	414
45	Vermont	647464	82
46	Virginia	8715698	989
47	Washington	7812880	1036
48	West Virginia	1770071	247
49	Wisconsin	5910955	654
50	Wyoming	584057	110

```
In [236... normalized_raw_data['Births per 1000 People'] = (normalized_raw_data['Birth Count'] /
```

```
In [237... print(normalized_raw_data)
```

	State of Residence	2023 Birth Count	Births per 1000 People
0	Alabama	5108468	637
1	Alaska	733406	209
2	Arizona	7431344	886
3	Arkansas	3067732	477
4	California	38965193	2417
5	Colorado	5877610	672
6	Connecticut	3617176	449
7	Delaware	1031890	229
8	District of Columbia	678972	198
9	Florida	22610726	1532
10	Georgia	11029227	1217
11	Hawaii	1435138	358
12	Idaho	1964726	254
13	Illinois	12549689	1118
14	Indiana	6862199	767
15	Iowa	3207004	412
16	Kansas	2940546	373
17	Kentucky	4526154	579
18	Louisiana	4573749	693
19	Maine	1395722	179
20	Maryland	6180253	867
21	Massachusetts	7001399	701
22	Michigan	10037261	982
23	Minnesota	5737915	704
24	Mississippi	2939690	505
25	Missouri	6196156	718
26	Montana	1132812	183
27	Nebraska	1978379	332
28	Nevada	3194176	519
29	New Hampshire	1402054	168
30	New Jersey	9290841	999
31	New Mexico	2114371	357
32	New York	19571216	1643
33	North Carolina	10835491	1199
34	North Dakota	783926	162
35	Ohio	11785935	1090
36	Oklahoma	4053824	625
37	Oregon	4233358	500
38	Pennsylvania	12961683	1160
39	Rhode Island	1095962	175
40	South Carolina	5373555	701
41	South Dakota	919318	199
42	Tennessee	7126489	820
43	Texas	30503301	2096
44	Utah	3417734	414
45	Vermont	647464	82
46	Virginia	8715698	989
47	Washington	7812880	1036
48	West Virginia	1770071	247
49	Wisconsin	5910955	654
50	Wyoming	584057	110

```
In [321]: def get_label_rotation(angle, offset):
# Rotation must be specified in degrees :(
rotation = np.rad2deg(angle + offset)
if angle <= np.pi:
    alignment = "right"
    rotation = rotation + 180
else:
    alignment = "left"
return rotation, alignment

ANGLES = np.linspace(0, 2 * np.pi, len(normalized_raw_data), endpoint=False)
VALUES = normalized_raw_data['Births per 1000 People'].values * 2
```

```

LABELS = normalized_raw_data['State of Residence'].values
# Convert Births Per 1000 Women to percentages
percentages = VALUES * 100

# Determine the width of each bar.
# The circumference is '2 * pi', so we divide that total width over the number of bar
WIDTH = 2 * np.pi / len(VALUES)

# Determines where to place the first bar.
# By default, matplotlib starts at 0 (the first bar is horizontal)
# but here we say we want to start at pi/2 (90 deg)
OFFSET = np.pi / 2

# Initialize Figure and Axis
fig, ax = plt.subplots(figsize=(50, 43), subplot_kw={"projection": "polar"})

# Specify offset
ax.set_theta_offset(OFFSET)

# Set limits for radial (y) axis. The negative lower bound creates the whole in the m
ax.set_ylim(-100, 100)

# Remove all spines
ax.set_frame_on(False)

# Remove grid and tick marks
ax.xaxis.grid(False)
ax.yaxis.grid(False)
ax.set_xticks([])
ax.set_yticks([])

# Add labels
def add_labels(angles, values, labels, offset, ax):

    # This is the space between the end of the bar and the label
    padding = 10 # Change this value for different padding

    # Iterate over angles, values, and labels, to add all of them.
    for angle, value, label, in zip(angles, values, labels):
        angle = angle

        # Obtain text rotation and alignment
        rotation, alignment = get_label_rotation(angle, offset)

        # And finally add the text
        ax.text(
            x=angle,
            y=value + padding,
            s=label,
            ha=alignment,
            va="center",
            rotation=rotation,
            rotation_mode="anchor",
            fontweight='bold',
            fontsize=45
        )

# Add labels
add_labels(ANGLES, percentages, LABELS, OFFSET, ax)

# Add bars with colormap
bars = ax.bar(
    ANGLES, percentages, width=WIDTH, linewidth=2,
    color=plt.cm.pink(percentages / 100), edgecolor="white"
)

```

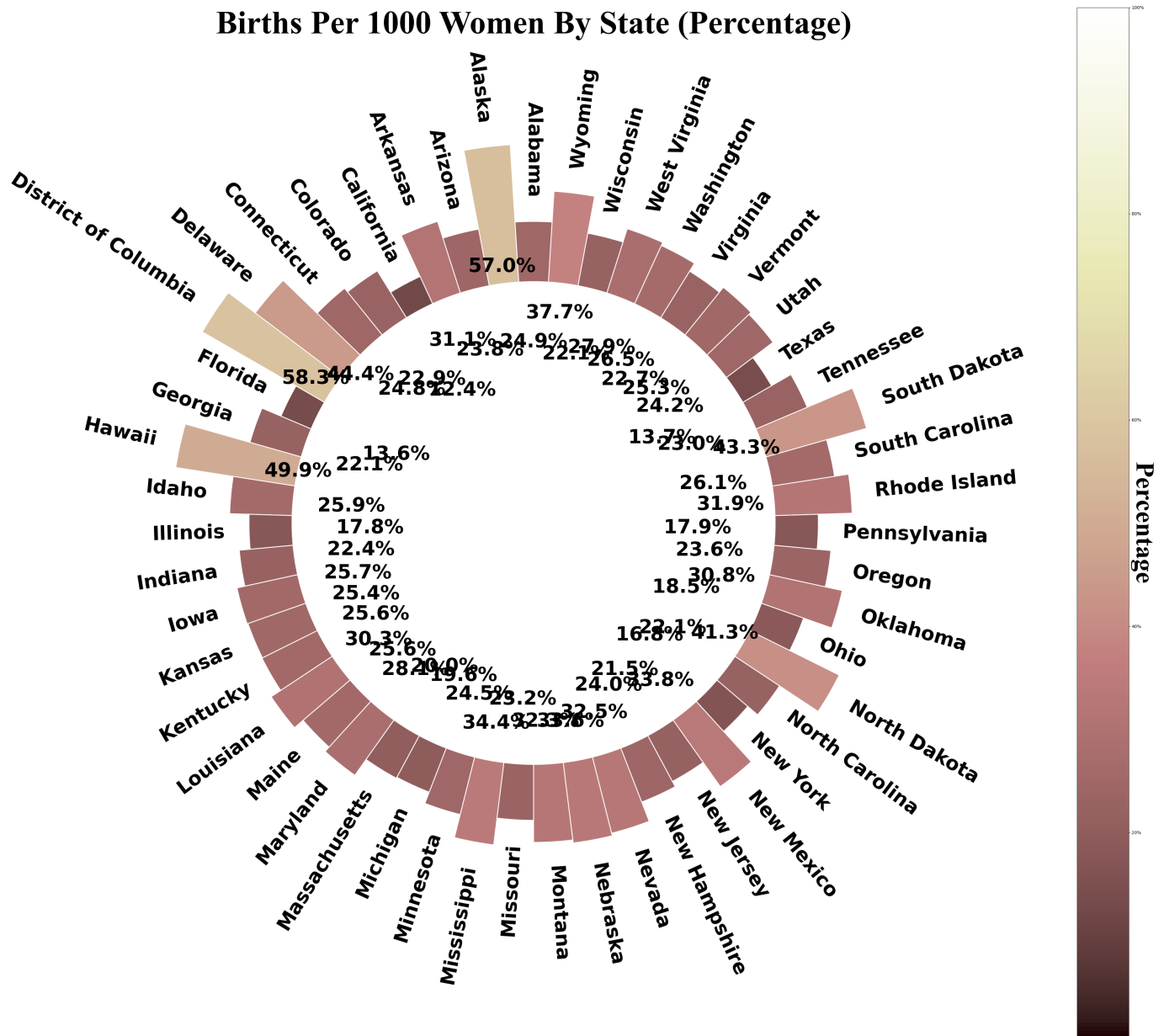
```

# Add percentages as text inside the circle
percentage_padding = -50 # Adjust this value for the desired padding
for angle, percentage in zip(ANGLES, percentages):
    ax.text(
        x=angle,
        y=percentage + percentage_padding,
        s=f'{percentage:.1f}%',
        ha='center',
        va='center',
        fontsize=45,
        fontweight='bold',
        color='black'
    )

# Add colorbar legend
cbar = plt.colorbar(plt.cm.ScalarMappable(cmap='pink', norm=plt.Normalize(vmin=0, vma
cbar.set_label('Percentage', rotation=270, labelpad=20, fontsize=60, fontweight='bol

plt.title('Births Per 1000 Women By State (Percentage)', fontsize=75, fontweight='bol
plt.savefig('births_of_mothers_by_state.png', transparent=True)
plt.show()

```



```

In [239... midpoint_mapping = {
    'No visits': 0,
    '1 to 2 visits': 1.5,
    '3 to 4 visits': 3.5,
    '5 to 6 visits': 5.5,

```

```
'7 to 8 visits': 7.5,  
'9 to 10 visits': 9.5,  
'11 to 12 visits': 11.5,  
'13 to 14 visits': 13.5,  
'15 to 16 visits': 15.5,  
'17 to 18 visits': 17.5,  
'19 or more visits': 19.5,  
'Unknown or Not Stated': 20.5}
```

```
In [240... state_avg_prenatal_visits = (raw_data.groupby('State of Residence')['Number of Prenat  
                                .apply(lambda x: x.map(midpoint_mapping).mean())  
                                .round(1)  
                                .rename("Average Number of Prenatal Visits"))
```

```
In [241... state_avg_month_prenatal_began = (raw_data.groupby('State of Residence')['Month Prena  
                                .apply(lambda x: x.mean()).round(1).rename("Average
```

```
In [242... state_avg_gestational_age = raw_data.groupby('State of Residence')['OE Gestational Ag
```

```
In [243... APNCU_index_categories_df = pd.concat([  
    state_avg_prenatal_visits,  
    state_avg_month_prenatal_began,  
    state_avg_gestational_age  
], axis=1).reset_index()
```

```
In [244... print(APNCU_index_categories_df)
```

	State of Residence	Average Number of Prenatal Visits \
0	Alabama	9.2
1	Alaska	10.7
2	Arizona	9.9
3	Arkansas	8.9
4	California	10.9
5	Colorado	9.6
6	Connecticut	11.4
7	Delaware	10.8
8	District of Columbia	9.8
9	Florida	9.4
10	Georgia	10.1
11	Hawaii	10.6
12	Idaho	10.7
13	Illinois	9.9
14	Indiana	9.8
15	Iowa	10.2
16	Kansas	9.7
17	Kentucky	9.8
18	Louisiana	10.0
19	Maine	11.4
20	Maryland	10.2
21	Massachusetts	10.4
22	Michigan	10.1
23	Minnesota	10.2
24	Mississippi	10.2
25	Missouri	9.5
26	Montana	10.8
27	Nebraska	10.7
28	Nevada	11.0
29	New Hampshire	11.3
30	New Jersey	9.3
31	New Mexico	9.4
32	New York	10.1
33	North Carolina	10.4
34	North Dakota	10.7
35	Ohio	10.0
36	Oklahoma	9.7
37	Oregon	10.9
38	Pennsylvania	10.1
39	Rhode Island	11.7
40	South Carolina	10.2
41	South Dakota	9.3
42	Tennessee	9.7
43	Texas	9.6
44	Utah	10.0
45	Vermont	12.0
46	Virginia	10.3
47	Washington	10.7
48	West Virginia	11.1
49	Wisconsin	10.6
50	Wyoming	11.4

	Average Month Prenatal Care Began	Average OE Gestational Age Weekly Code
0	4.7	37.3
1	4.4	38.7
2	6.9	38.1
3	5.1	38.0
4	7.1	37.5
5	5.5	38.0
6	4.6	38.1
7	7.8	38.5
8	7.1	38.7
9	8.7	37.1
10	5.3	37.3

11	9.2	38.7
12	4.1	38.2
13	6.0	37.6
14	3.9	37.7
15	3.7	38.1
16	3.8	38.1
17	4.8	37.7
18	5.8	37.7
19	4.1	38.4
20	8.9	38.1
21	5.9	38.7
22	7.4	37.8
23	4.6	38.3
24	6.8	37.7
25	6.1	37.8
26	3.1	38.4
27	5.7	38.0
28	5.5	38.1
29	3.6	38.7
30	5.1	37.9
31	7.9	38.3
32	6.5	37.7
33	4.7	37.7
34	5.5	38.3
35	5.4	37.5
36	6.0	38.2
37	4.3	38.5
38	7.2	38.0
39	3.4	38.6
40	4.5	37.7
41	3.1	38.3
42	6.0	37.7
43	8.1	36.8
44	5.7	37.9
45	2.6	38.5
46	5.9	37.9
47	9.3	38.5
48	4.3	37.8
49	6.0	38.1
50	6.5	38.4

```
In [245... print(APNCU_index_categories_df.columns)
```

```
Index(['State of Residence', 'Average Number of Prenatal Visits',
      'Average Month Prenatal Care Began',
      'Average OE Gestational Age Weekly Code'],
      dtype='object')
```

```
In [246... def calculate_apncu_index(row):
    gestational_age = round(row['Average OE Gestational Age Weekly Code'], 1)
    num_prenatal_visits = round(row['Average Number of Prenatal Visits'], 1)
    month_began = round(row['Average Month Prenatal Care Began'], 1)

    if month_began <= 4.4:
        if (gestational_age <= 13.4 and num_prenatal_visits >= 1.4) or \
            (14.4 <= gestational_age <= 17.4 and num_prenatal_visits >= 2.4) or \
            (18.4 <= gestational_age <= 21.4 and num_prenatal_visits >= 2.4) or \
            (22.4 <= gestational_age <= 25.4 and num_prenatal_visits >= 3.4) or \
            (26.4 <= gestational_age <= 29.4 and num_prenatal_visits >= 3.4) or \
            (30.4 <= gestational_age <= 31.4 and num_prenatal_visits >= 4.4) or \
            (32.4 <= gestational_age <= 33.4 and num_prenatal_visits >= 4.4) or \
            (34.4 <= gestational_age <= 40.4 and 5.4 <= num_prenatal_visits <= 9.4):
            return 'Adequate'
        elif (34.5 <= gestational_age <= 40.4 and 9.5 <= num_prenatal_visits <= 12.4):
            return 'Adequate Plus'
```

```

    if month_began >= 6.5:
        if (14.4 <= gestational_age <= 21.4 and num_prenatal_visits == 0) or \
            (22.4 <= gestational_age <= 29.4 and num_prenatal_visits <= 1.4) or \
            (30.4 <= gestational_age <= 31.4 and num_prenatal_visits <= 2.4) or \
            (32.4 <= gestational_age <= 33.4 and num_prenatal_visits <= 3.4) or \
            (33.5 <= gestational_age <= 34.4 and num_prenatal_visits <= 4.4) or \
            (34.5 <= gestational_age <= 40.4 and 4.5 <= num_prenatal_visits <= 12.4) :
            return 'Inadequate'

    if 4.5 <= month_began <= 6.4:
        return 'Intermediate'

```

In [247... APNCU_index_categories_df['APNCU Score'] = APNCU_index_categories_df.apply(calculate_

In [248... print(APNCU_index_categories_df)

	State of Residence	Average Number of Prenatal Visits \
0	Alabama	9.2
1	Alaska	10.7
2	Arizona	9.9
3	Arkansas	8.9
4	California	10.9
5	Colorado	9.6
6	Connecticut	11.4
7	Delaware	10.8
8	District of Columbia	9.8
9	Florida	9.4
10	Georgia	10.1
11	Hawaii	10.6
12	Idaho	10.7
13	Illinois	9.9
14	Indiana	9.8
15	Iowa	10.2
16	Kansas	9.7
17	Kentucky	9.8
18	Louisiana	10.0
19	Maine	11.4
20	Maryland	10.2
21	Massachusetts	10.4
22	Michigan	10.1
23	Minnesota	10.2
24	Mississippi	10.2
25	Missouri	9.5
26	Montana	10.8
27	Nebraska	10.7
28	Nevada	11.0
29	New Hampshire	11.3
30	New Jersey	9.3
31	New Mexico	9.4
32	New York	10.1
33	North Carolina	10.4
34	North Dakota	10.7
35	Ohio	10.0
36	Oklahoma	9.7
37	Oregon	10.9
38	Pennsylvania	10.1
39	Rhode Island	11.7
40	South Carolina	10.2
41	South Dakota	9.3
42	Tennessee	9.7
43	Texas	9.6
44	Utah	10.0
45	Vermont	12.0
46	Virginia	10.3
47	Washington	10.7
48	West Virginia	11.1
49	Wisconsin	10.6
50	Wyoming	11.4

	Average Month Prenatal Care Began	Average OE Gestational Age Weekly Code \
0	4.7	37.3
1	4.4	38.7
2	6.9	38.1
3	5.1	38.0
4	7.1	37.5
5	5.5	38.0
6	4.6	38.1
7	7.8	38.5
8	7.1	38.7
9	8.7	37.1
10	5.3	37.3

11	9.2	38.7
12	4.1	38.2
13	6.0	37.6
14	3.9	37.7
15	3.7	38.1
16	3.8	38.1
17	4.8	37.7
18	5.8	37.7
19	4.1	38.4
20	8.9	38.1
21	5.9	38.7
22	7.4	37.8
23	4.6	38.3
24	6.8	37.7
25	6.1	37.8
26	3.1	38.4
27	5.7	38.0
28	5.5	38.1
29	3.6	38.7
30	5.1	37.9
31	7.9	38.3
32	6.5	37.7
33	4.7	37.7
34	5.5	38.3
35	5.4	37.5
36	6.0	38.2
37	4.3	38.5
38	7.2	38.0
39	3.4	38.6
40	4.5	37.7
41	3.1	38.3
42	6.0	37.7
43	8.1	36.8
44	5.7	37.9
45	2.6	38.5
46	5.9	37.9
47	9.3	38.5
48	4.3	37.8
49	6.0	38.1
50	6.5	38.4

	APNCU Score
0	Intermediate
1	Adequate Plus
2	Inadequate
3	Intermediate
4	Inadequate
5	Intermediate
6	Intermediate
7	Inadequate
8	Inadequate
9	Inadequate
10	Intermediate
11	Inadequate
12	Adequate Plus
13	Intermediate
14	Adequate Plus
15	Adequate Plus
16	Adequate Plus
17	Intermediate
18	Intermediate
19	Adequate Plus
20	Inadequate
21	Intermediate
22	Inadequate

```

23 Intermediate
24 Inadequate
25 Intermediate
26 Adequate Plus
27 Intermediate
28 Intermediate
29 Adequate Plus
30 Intermediate
31 Inadequate
32 Inadequate
33 Intermediate
34 Intermediate
35 Intermediate
36 Intermediate
37 Adequate Plus
38 Inadequate
39 Adequate Plus
40 Intermediate
41 Adequate
42 Intermediate
43 Inadequate
44 Intermediate
45 Adequate Plus
46 Intermediate
47 Inadequate
48 Adequate Plus
49 Intermediate
50 Inadequate

```

```
In [249... APNCU_categories_counts = APNCU_index_categories_df['APNCU Score'].value_counts()
print(APNCU_categories_counts)
```

```

APNCU Score
Intermediate      23
Inadequate        15
Adequate Plus     12
Adequate          1
Name: count, dtype: int64

```

```
In [250... folder_path = '//Users/keriwilkins/Downloads/tl_2023_us_state'
```

```
In [251... gdf = gpd.read_file(folder_path)
```

```
In [252... print(gdf.head())
```

	REGION	DIVISION	STATEFP	STATENS	GEOID	GEOIDFQ	STUSPS	NAME	\
0	3	5	54	01779805	54	0400000US54	WV	West Virginia	
1	3	5	12	00294478	12	0400000US12	FL	Florida	
2	2	3	17	01779784	17	0400000US17	IL	Illinois	
3	2	4	27	00662849	27	0400000US27	MN	Minnesota	
4	3	5	24	01714934	24	0400000US24	MD	Maryland	

	LSAD	MTFCC	FUNCSTAT	ALAND	AWATER	INTPTLAT	INTPTLON	\
0	00	G4000	A	62266499712	489003081	+38.6472854	-080.6183274	
1	00	G4000	A	138963763779	45970528648	+28.3989775	-082.5143005	
2	00	G4000	A	143778366814	6216688589	+40.1028754	-089.1526108	
3	00	G4000	A	206244555303	18937471947	+46.3159573	-094.1996043	
4	00	G4000	A	25151736098	6979330958	+38.9466584	-076.6744939	

```

                                geometry
0  POLYGON ((-77.75438 39.33346, -77.75422 39.333...
1  MULTIPOLYGON (((-83.10874 24.62949, -83.10711 ...
2  POLYGON ((-87.89243 38.28285, -87.89334 38.282...
3  POLYGON ((-95.31991 48.99892, -95.31778 48.998...
4  POLYGON ((-75.75600 39.24607, -75.75579 39.243...

```

```
In [253... gdf = gdf.merge(APNCU_index_categories_df, left_on='NAME', right_on='State of Residence
```

```
In [254... print(gdf)
```

	REGION	DIVISION	STATEFP	STATENS	GEOID	GEOIDFQ	STUSPS	\
0	3	5	54	01779805	54	0400000US54	WV	
1	3	5	12	00294478	12	0400000US12	FL	
2	2	3	17	01779784	17	0400000US17	IL	
3	2	4	27	00662849	27	0400000US27	MN	
4	3	5	24	01714934	24	0400000US24	MD	
5	1	1	44	01219835	44	0400000US44	RI	
6	4	8	16	01779783	16	0400000US16	ID	
7	1	1	33	01779794	33	0400000US33	NH	
8	3	5	37	01027616	37	0400000US37	NC	
9	1	1	50	01779802	50	0400000US50	VT	
10	1	1	09	01779780	09	0400000US09	CT	
11	3	5	10	01779781	10	0400000US10	DE	
12	4	8	35	00897535	35	0400000US35	NM	
13	4	9	06	01779778	06	0400000US06	CA	
14	1	2	34	01779795	34	0400000US34	NJ	
15	2	3	55	01779806	55	0400000US55	WI	
16	4	9	41	01155107	41	0400000US41	OR	
17	2	4	31	01779792	31	0400000US31	NE	
18	1	2	42	01779798	42	0400000US42	PA	
19	4	9	53	01779804	53	0400000US53	WA	
20	3	7	22	01629543	22	0400000US22	LA	
21	3	5	13	01705317	13	0400000US13	GA	
22	3	6	01	01779775	01	0400000US01	AL	
23	4	8	49	01455989	49	0400000US49	UT	
24	2	3	39	01085497	39	0400000US39	OH	
25	3	7	48	01779801	48	0400000US48	TX	
26	4	8	08	01779779	08	0400000US08	CO	
27	3	5	45	01779799	45	0400000US45	SC	
28	3	7	40	01102857	40	0400000US40	OK	
29	3	6	47	01325873	47	0400000US47	TN	
30	4	8	56	01779807	56	0400000US56	WY	
31	4	9	15	01779782	15	0400000US15	HI	
32	2	4	38	01779797	38	0400000US38	ND	
33	3	6	21	01779786	21	0400000US21	KY	
34	1	1	23	01779787	23	0400000US23	ME	
35	1	2	36	01779796	36	0400000US36	NY	
36	4	8	32	01779793	32	0400000US32	NV	
37	4	9	02	01785533	02	0400000US02	AK	
38	2	3	26	01779789	26	0400000US26	MI	
39	3	7	05	00068085	05	0400000US05	AR	
40	3	6	28	01779790	28	0400000US28	MS	
41	2	4	29	01779791	29	0400000US29	MO	
42	4	8	30	00767982	30	0400000US30	MT	
43	2	4	20	00481813	20	0400000US20	KS	
44	2	3	18	00448508	18	0400000US18	IN	
45	2	4	46	01785534	46	0400000US46	SD	
46	1	1	25	00606926	25	0400000US25	MA	
47	3	5	51	01779803	51	0400000US51	VA	
48	3	5	11	01702382	11	0400000US11	DC	
49	2	4	19	01779785	19	0400000US19	IA	
50	4	8	04	01779777	04	0400000US04	AZ	

		NAME	LSAD	MTFCC	...	ALAND	AWATER	\
0		West Virginia	00	G4000	...	62266499712	489003081	
1		Florida	00	G4000	...	138963763779	45970528648	
2		Illinois	00	G4000	...	143778366814	6216688589	
3		Minnesota	00	G4000	...	206244555303	18937471947	
4		Maryland	00	G4000	...	25151736098	6979330958	
5		Rhode Island	00	G4000	...	2677763372	1323686976	
6		Idaho	00	G4000	...	214049886849	2391614331	
7		New Hampshire	00	G4000	...	23190126365	1025956733	
8		North Carolina	00	G4000	...	125935880061	13453540851	
9		Vermont	00	G4000	...	23872589127	1030648383	
10		Connecticut	00	G4000	...	12541750274	1816364426	

11	Delaware	00	G4000	...	5046703781	1399207462
12	New Mexico	00	G4000	...	314198587197	726463919
13	California	00	G4000	...	403673296401	20291770234
14	New Jersey	00	G4000	...	19049183398	3532816229
15	Wisconsin	00	G4000	...	140292627460	29343084365
16	Oregon	00	G4000	...	248630419895	6168960338
17	Nebraska	00	G4000	...	198949602728	1379309601
18	Pennsylvania	00	G4000	...	115881839569	3397855215
19	Washington	00	G4000	...	172118778986	12548923076
20	Louisiana	00	G4000	...	111930452904	23721187320
21	Georgia	00	G4000	...	149485311347	4419673221
22	Alabama	00	G4000	...	131185049346	4582326383
23	Utah	00	G4000	...	213921849163	5963196691
24	Ohio	00	G4000	...	105823831336	10274524796
25	Texas	00	G4000	...	676686238592	18982083586
26	Colorado	00	G4000	...	268418756810	1185758065
27	South Carolina	00	G4000	...	77866000436	5074443652
28	Oklahoma	00	G4000	...	177664484361	3373395450
29	Tennessee	00	G4000	...	106792311478	2322248149
30	Wyoming	00	G4000	...	251458162746	1868053273
31	Hawaii	00	G4000	...	16634423916	11777375352
32	North Dakota	00	G4000	...	178694342677	4414747947
33	Kentucky	00	G4000	...	102266598312	2384223544
34	Maine	00	G4000	...	79888284131	11745177995
35	New York	00	G4000	...	122049078273	19256833049
36	Nevada	00	G4000	...	284537045712	1839880833
37	Alaska	00	G4000	...	1479016910296	245347100126
38	Michigan	00	G4000	...	146619947556	103866186527
39	Arkansas	00	G4000	...	134660466558	3122251184
40	Mississippi	00	G4000	...	121533540877	3914738613
41	Missouri	00	G4000	...	178052260322	2487519141
42	Montana	00	G4000	...	376973185941	3867177570
43	Kansas	00	G4000	...	211753777631	1345707708
44	Indiana	00	G4000	...	92786613552	1543998356
45	South Dakota	00	G4000	...	196341525171	3387709166
46	Massachusetts	00	G4000	...	20204345054	7130705555
47	Virginia	00	G4000	...	102258163252	8528087616
48	District of Columbia	00	G4000	...	158316184	18709787
49	Iowa	00	G4000	...	144659283871	1086402333
50	Arizona	00	G4000	...	294366106734	854003932

	INTPTLAT	INTPTLON	\
0	+38.6472854	-080.6183274	
1	+28.3989775	-082.5143005	
2	+40.1028754	-089.1526108	
3	+46.3159573	-094.1996043	
4	+38.9466584	-076.6744939	
5	+41.5964850	-071.5264901	
6	+44.3484222	-114.5588538	
7	+43.6727945	-071.5841886	
8	+35.5397100	-079.1308636	
9	+44.0589536	-072.6710173	
10	+41.5798637	-072.7466572	
11	+38.9985661	-075.4416440	
12	+34.4346843	-106.1316181	
13	+37.1551773	-119.5434183	
14	+40.1072744	-074.6652012	
15	+44.6309071	-089.7093916	
16	+43.9717125	-120.6229578	
17	+41.5433053	-099.8118646	
18	+40.9046042	-077.8275233	
19	+47.4073238	-120.5757999	
20	+30.8634368	-091.7987173	
21	+32.6295789	-083.4235109	
22	+32.7395785	-086.8434469	

23 +39.3349925 -111.6563326
 24 +40.4149297 -082.7119975
 25 +31.4347032 -099.2818238
 26 +38.9937669 -105.5087122
 27 +33.8741776 -080.8542639
 28 +35.5900815 -097.4867789
 29 +35.8584600 -086.3496339
 30 +42.9896591 -107.5443922
 31 +19.8281671 -155.4950421
 32 +47.4421740 -100.4608258
 33 +37.5336844 -085.2929801
 34 +45.4092843 -068.6666160
 35 +42.9133974 -075.5962723
 36 +39.3310928 -116.6151469
 37 +63.3473560 -152.8397334
 38 +44.8441768 -085.6604907
 39 +34.8955256 -092.4446262
 40 +32.6864714 -089.6561377
 41 +38.3507500 -092.4567826
 42 +47.0511770 -109.6348174
 43 +38.4985464 -098.3834298
 44 +39.9013136 -086.2919129
 45 +44.4467957 -100.2381762
 46 +42.1565196 -071.4895915
 47 +37.5222512 -078.6681938
 48 +38.9042474 -077.0165167
 49 +42.0700243 -093.4933473
 50 +34.2039362 -111.6063449

	geometry	State of Residence \
0	POLYGON ((-77.75438 39.33346, -77.75422 39.333...	West Virginia
1	MULTIPOLYGON (((-83.10874 24.62949, -83.10711 ...	Florida
2	POLYGON ((-87.89243 38.28285, -87.89334 38.282...	Illinois
3	POLYGON ((-95.31991 48.99892, -95.31778 48.998...	Minnesota
4	POLYGON ((-75.75600 39.24607, -75.75579 39.243...	Maryland
5	MULTIPOLYGON (((-71.67881 41.15891, -71.67626 ...	Rhode Island
6	POLYGON ((-116.35844 49.00047, -116.34805 49.0...	Idaho
7	POLYGON ((-70.83887 43.24449, -70.83886 43.244...	New Hampshire
8	POLYGON ((-77.89977 36.54460, -77.89942 36.544...	North Carolina
9	POLYGON ((-72.04187 44.15665, -72.04180 44.156...	Vermont
10	POLYGON ((-72.52790 41.17777, -72.55156 41.173...	Connecticut
11	POLYGON ((-75.50949 39.68652, -75.50942 39.686...	Delaware
12	POLYGON ((-103.06472 32.33465, -103.06470 32.3...	New Mexico
13	MULTIPOLYGON (((-119.99987 41.18397, -119.9998...	California
14	POLYGON ((-74.46493 39.29823, -74.46906 39.294...	New Jersey
15	POLYGON ((-89.55482 46.22790, -89.55480 46.227...	Wisconsin
16	POLYGON ((-121.44123 45.69728, -121.42442 45.6...	Oregon
17	POLYGON ((-95.93779 41.39075, -95.93742 41.389...	Nebraska
18	POLYGON ((-75.59310 39.83772, -75.59319 39.837...	Pennsylvania
19	POLYGON ((-118.40382 46.00077, -118.40506 46.0...	Washington
20	POLYGON ((-92.72474 33.01439, -92.72365 33.014...	Louisiana
21	POLYGON ((-81.09538 31.52098, -81.09710 31.519...	Georgia
22	POLYGON ((-85.12733 31.76256, -85.12753 31.762...	Alabama
23	POLYGON ((-111.50781 41.99955, -111.50727 41.9...	Utah
24	POLYGON ((-80.51915 40.90033, -80.51914 40.899...	Ohio
25	POLYGON ((-98.42353 34.08284, -98.42235 34.082...	Texas
26	POLYGON ((-105.15504 36.99526, -105.15543 36.9...	Colorado
27	POLYGON ((-81.36660 35.16488, -81.35786 35.164...	South Carolina
28	POLYGON ((-95.78676 36.99931, -95.78601 36.999...	Oklahoma
29	POLYGON ((-83.98762 36.58947, -83.98724 36.589...	Tennessee
30	POLYGON ((-106.32116 40.99912, -106.32621 40.9...	Wyoming
31	MULTIPOLYGON (((-164.76481 23.58172, -164.7634...	Hawaii
32	POLYGON ((-98.99982 48.99996, -98.93769 49.000...	North Dakota
33	MULTIPOLYGON (((-89.57120 36.55214, -89.57114 ...	Kentucky
34	POLYGON ((-67.80569 45.68005, -67.80363 45.677...	Maine

35	MULTIPOLYGON (((-74.72623 44.99486, -74.72257 ...	New York
36	POLYGON ((-119.32418 41.99392, -119.32362 41.9...	Nevada
37	MULTIPOLYGON (((-179.23023 51.28393, -179.2256...	Alaska
38	POLYGON ((-87.11270 47.81800, -87.06310 47.799...	Michigan
39	POLYGON ((-90.95577 34.11871, -90.95451 34.117...	Arkansas
40	POLYGON ((-88.43138 32.22767, -88.43180 32.224...	Mississippi
41	POLYGON ((-93.37439 40.58033, -93.37429 40.580...	Missouri
42	POLYGON ((-105.05764 48.99923, -105.03410 48.9...	Montana
43	POLYGON ((-94.61764 37.76124, -94.61766 37.760...	Kansas
44	POLYGON ((-86.32960 38.18180, -86.33037 38.182...	Indiana
45	POLYGON ((-98.72437 45.93873, -98.72425 45.938...	South Dakota
46	POLYGON ((-71.25515 42.73655, -71.25510 42.736...	Massachusetts
47	POLYGON ((-76.49150 36.55065, -76.49388 36.550...	Virginia
48	POLYGON ((-77.11975 38.93435, -77.11886 38.935...	District of Columbia
49	POLYGON ((-95.86095 43.50004, -95.85979 43.500...	Iowa
50	POLYGON ((-110.75069 37.00301, -110.74193 37.0...	Arizona

	Average Number of Prenatal Visits	Average Month Prenatal Care Began \
0	11.1	4.3
1	9.4	8.7
2	9.9	6.0
3	10.2	4.6
4	10.2	8.9
5	11.7	3.4
6	10.7	4.1
7	11.3	3.6
8	10.4	4.7
9	12.0	2.6
10	11.4	4.6
11	10.8	7.8
12	9.4	7.9
13	10.9	7.1
14	9.3	5.1
15	10.6	6.0
16	10.9	4.3
17	10.7	5.7
18	10.1	7.2
19	10.7	9.3
20	10.0	5.8
21	10.1	5.3
22	9.2	4.7
23	10.0	5.7
24	10.0	5.4
25	9.6	8.1
26	9.6	5.5
27	10.2	4.5
28	9.7	6.0
29	9.7	6.0
30	11.4	6.5
31	10.6	9.2
32	10.7	5.5
33	9.8	4.8
34	11.4	4.1
35	10.1	6.5
36	11.0	5.5
37	10.7	4.4
38	10.1	7.4
39	8.9	5.1
40	10.2	6.8
41	9.5	6.1
42	10.8	3.1
43	9.7	3.8
44	9.8	3.9
45	9.3	3.1
46	10.4	5.9

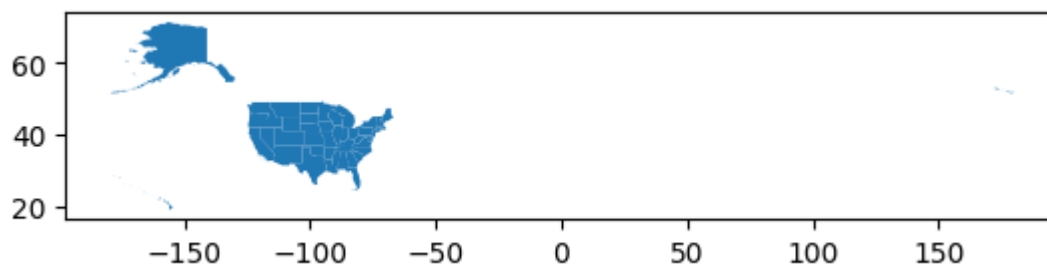
47	10.3	5.9
48	9.8	7.1
49	10.2	3.7
50	9.9	6.9

	Average OE Gestational Age Weekly	Code	APNCU Score
0	37.8	Adequate Plus	
1	37.1	Inadequate	
2	37.6	Intermediate	
3	38.3	Intermediate	
4	38.1	Inadequate	
5	38.6	Adequate Plus	
6	38.2	Adequate Plus	
7	38.7	Adequate Plus	
8	37.7	Intermediate	
9	38.5	Adequate Plus	
10	38.1	Intermediate	
11	38.5	Inadequate	
12	38.3	Inadequate	
13	37.5	Inadequate	
14	37.9	Intermediate	
15	38.1	Intermediate	
16	38.5	Adequate Plus	
17	38.0	Intermediate	
18	38.0	Inadequate	
19	38.5	Inadequate	
20	37.7	Intermediate	
21	37.3	Intermediate	
22	37.3	Intermediate	
23	37.9	Intermediate	
24	37.5	Intermediate	
25	36.8	Inadequate	
26	38.0	Intermediate	
27	37.7	Intermediate	
28	38.2	Intermediate	
29	37.7	Intermediate	
30	38.4	Inadequate	
31	38.7	Inadequate	
32	38.3	Intermediate	
33	37.7	Intermediate	
34	38.4	Adequate Plus	
35	37.7	Inadequate	
36	38.1	Intermediate	
37	38.7	Adequate Plus	
38	37.8	Inadequate	
39	38.0	Intermediate	
40	37.7	Inadequate	
41	37.8	Intermediate	
42	38.4	Adequate Plus	
43	38.1	Adequate Plus	
44	37.7	Adequate Plus	
45	38.3	Adequate	
46	38.7	Intermediate	
47	37.9	Intermediate	
48	38.7	Inadequate	
49	38.1	Adequate Plus	
50	38.1	Inadequate	

[51 rows x 21 columns]

In [255... `gdf.plot()`

Out [255... `<Axes: >`



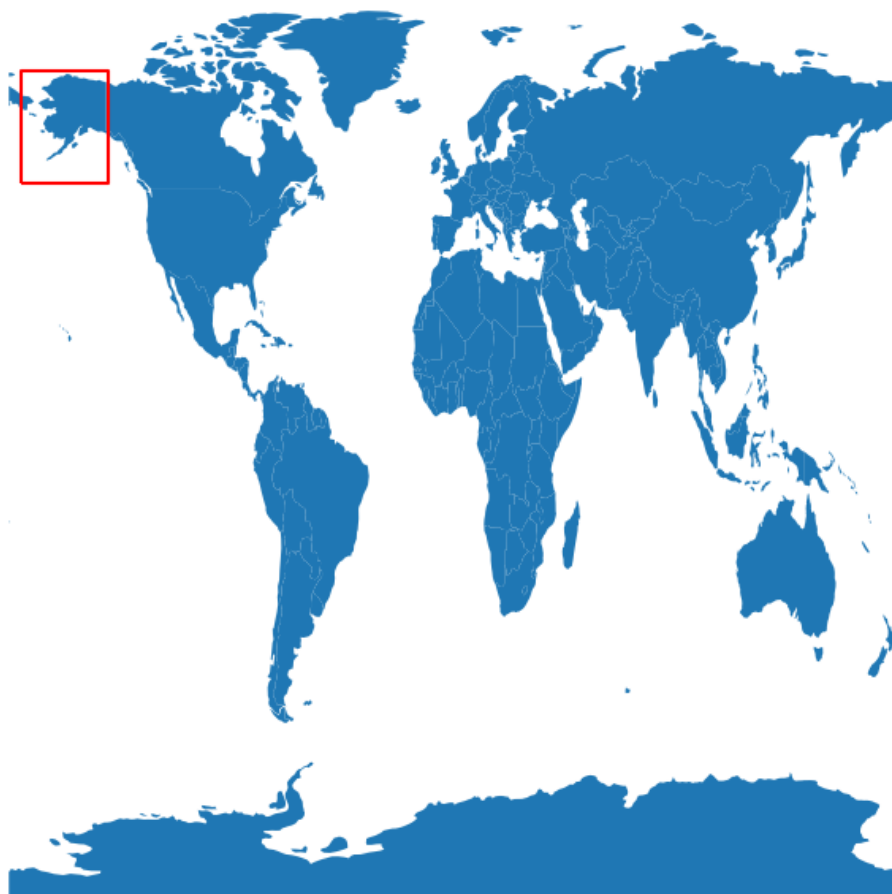
```
In [256...] world = gpd.read_file('/Users/keriwilkins/Downloads/ne_110m_admin_0_countries/ne_110m

# NOTE: the convention for polygon points is (Long, Lat)...counterintuitive
polygon = Polygon([(-175,50),(-175,72),(-140, 72),(-140,50)])
# polygon = Polygon([(-180,0),(-180,90),(-120,90),(-120,0)])

# polygon=hipolygon
poly_gdf = gpd.GeoDataFrame( geometry=[polygon], crs=world.crs)

fig, ax1 = plt.subplots(1, figsize=(8, 18))
world.plot(ax=ax1)
poly_gdf.boundary.plot(ax = ax1, color="red")
ax1.set_title("The red polygon can be used to clip Alaska's western islands", fontsize=12)
ax1.set_axis_off()
plt.show()
```

The red polygon can be used to clip Alaska's western islands

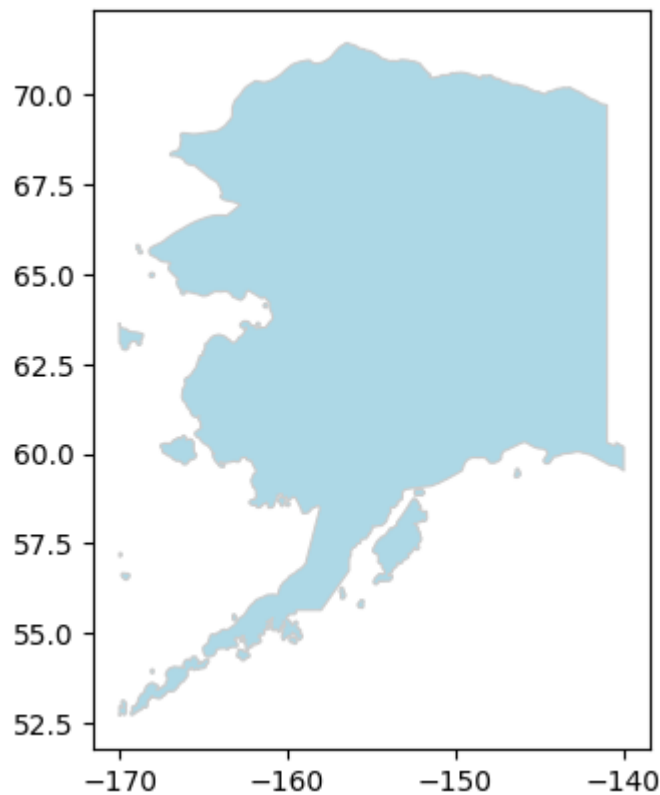


```
In [257...] # Define your polygon
polygon = Polygon([(-170,50),(-170,72),(-140, 72),(-140,50)])

# Apply clipping
clipped_alaska = gdf.clip(polygon)

# Plot the clipped geometry
clipped_alaska.plot(color='lightblue', linewidth=0.8, edgecolor='0.8')
```

Out [257... <Axes: >



```
In [258... category_mapping = {
    "Adequate Plus": 90,
    "Adequate": 70,
    "Intermediate": 50,
    "Inadequate": 30
}

# Map the categorical values to numerical values in the DataFrame column
gdf['APNCU Score Numerical'] = gdf['APNCU Score'].map(category_mapping)

# Check if the mapping was successful
print(gdf[['APNCU Score', 'APNCU Score Numerical']].head())
```

	APNCU Score	APNCU Score Numerical
0	Adequate Plus	90
1	Inadequate	30
2	Intermediate	50
3	Intermediate	50
4	Inadequate	30

```
In [259... gdf = gdf.to_crs(epsg=4326)
```

```
In [260... # Create a "copy" of gdf for re-projecting
gdf = gdf.to_crs('epsg:4326')

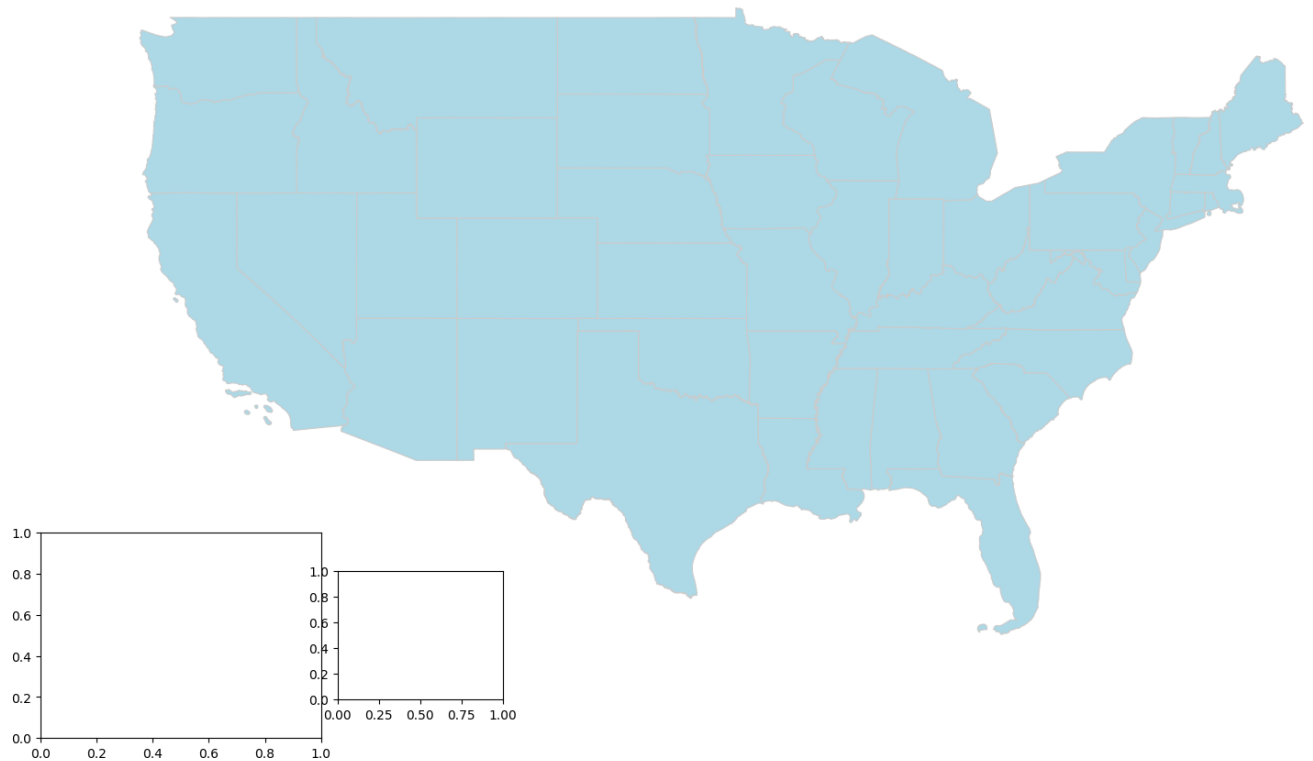
# create figure and axes for with Matplotlib for main map
fig, ax = plt.subplots(1, figsize=(18, 14))
# remove the axis box from the main map
ax.axis('off')

# create map of all states except AK and HI in the main map axis
gdf[~gdf.STUSPS.isin(['HI', 'AK'])].plot(color='lightblue', linewidth=0.8, ax=ax, edge

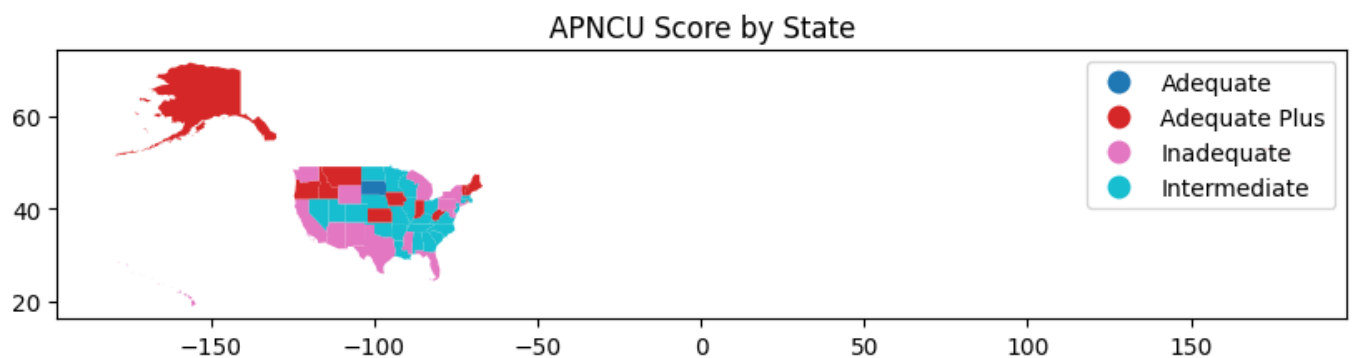
# Add Alaska Axis (x, y, width, height)
akax = fig.add_axes([0.1, 0.17, 0.17, 0.16])
```

```
# Add Hawaii Axis(x, y, width, height)
hiax = fig.add_axes([.28, 0.20, 0.1, 0.1])

# We'll later map Alaska in "akax" and Hawaii in "hiax"
```



```
In [261... gdf.plot(column='APNCU Score', legend=True, figsize=(10, 8))
ax.axis('off')
plt.title('APNCU Score by State')
plt.show()
```



```
In [262... fig, ax = plt.subplots(figsize=(18, 14))
ax.axis('off') # Remove the axis box from the main map

# Plot all states except Alaska and Hawaii in the main map axis
gdf[~gdf.STUSPS.isin(['HI', 'AK'])].plot(color='lightblue', linewidth=0.8, ax=ax, edge

# Add Alaska Axis (x, y, width, height)
akax = fig.add_axes([0.1, 0.17, 0.17, 0.16])

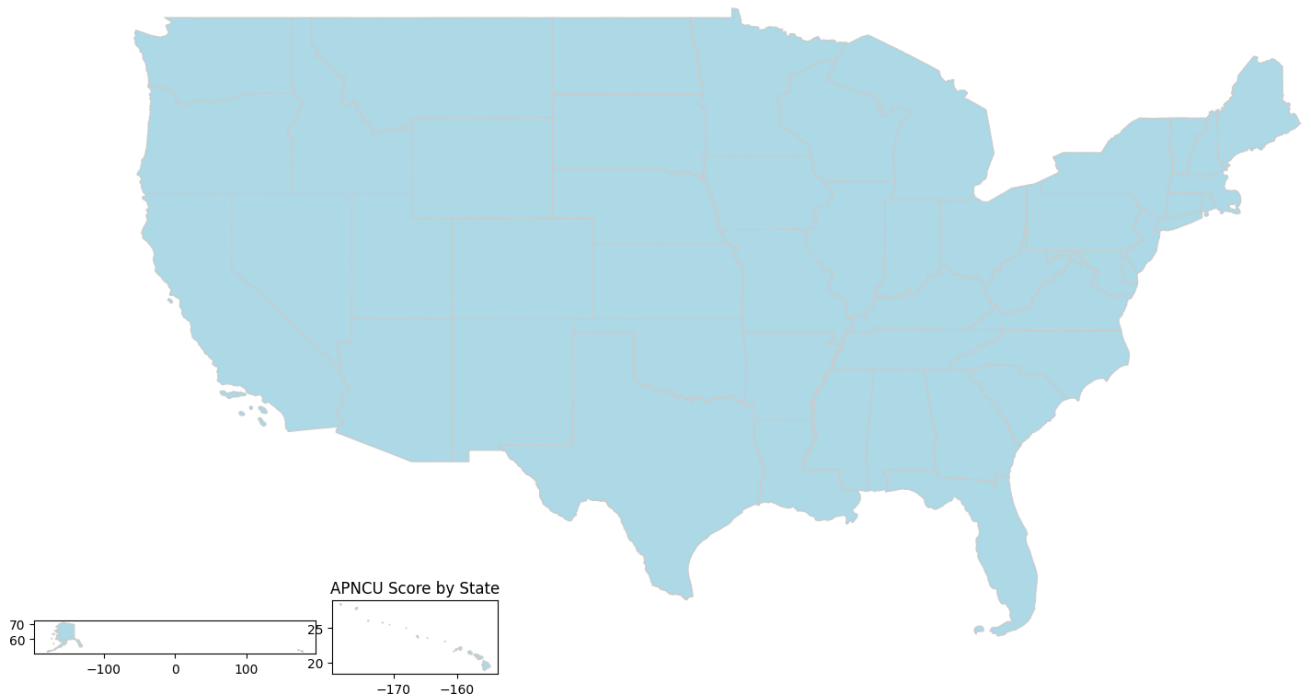
# Plot Alaska in the Alaska axis
gdf[gdf.STUSPS == 'AK'].plot(color='lightblue', linewidth=0.8, ax=akax, edgecolor='0.

# Add Hawaii Axis (x, y, width, height)
hiax = fig.add_axes([0.28, 0.20, 0.1, 0.1])

# Plot Hawaii in the Hawaii axis
gdf[gdf.STUSPS == 'HI'].plot(color='lightblue', linewidth=0.8, ax=hiax, edgecolor='0.
```

```
# Set the title
plt.title('APNCU Score by State')

# Show the plot
plt.show()
```



```
In [263... fig, ax = plt.subplots(figsize=(18, 14))
ax.axis('off') # Remove the axis box from the main map

# Plot all states except Alaska and Hawaii in the main map axis
gdf[~gdf.STUSPS.isin(['HI', 'AK'])].plot(color='lightblue', linewidth=0.8, ax=ax, edge

# Add Alaska Axis (x, y, width, height)
akax = fig.add_axes([0.1, 0.17, 0.17, 0.16])
akax.axis('off')

# Plot Alaska in the Alaska axis
gdf[gdf.STUSPS == 'AK'].plot(color='lightblue', linewidth=0.8, ax=akax, edgecolor='0.

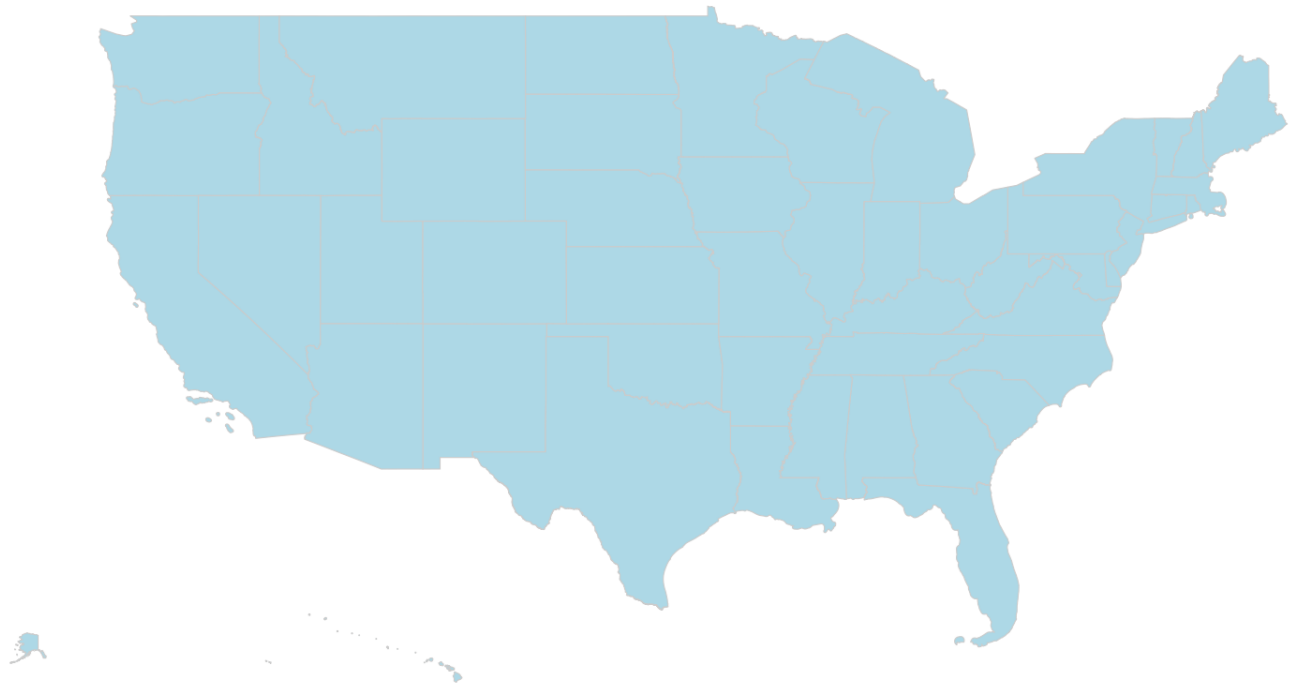
# Add Hawaii Axis (x, y, width, height)
hiax = fig.add_axes([0.1, 0.17, 0.17, 0.16])
hiax.axis('off')

# Plot Hawaii in the Hawaii axis
gdf[gdf.STUSPS == 'HI'].plot(color='lightblue', linewidth=0.8, ax=hiax, edgecolor='0.

# Adjust Hawaii position
hiax.set_position([0.28, 0.20, 0.1, 0.1])

# Set the title outside of the boxes
fig.suptitle('APNCU Score by State', y=0.95)

# Show the plot
plt.show()
```



```
In [322... fig, ax = plt.subplots(figsize=(30, 15))
ax.axis('off')# Remove the axis box from the main map

# Plot all states except Alaska and Hawaii in the main map axis with APNCU Score Nume
gdf[~gdf.STUSPS.isin(['HI', 'AK'])].plot(column='APNCU Score Numerical', cmap='pink',

# Add Alaska Axis (x, y, width, height) – Increase width and height
akax = fig.add_axes([0.2, 0.2, 0.4, 0.2])
akax.axis('off')

# Plot Alaska in the Alaska axis
gdf[gdf.STUSPS == 'AK'].plot(column='APNCU Score Numerical', cmap='pink', linewidth=0

# Add Hawaii Axis (x, y, width, height) – Increase width and height
hiax = fig.add_axes([0.28, 0.22, 0.2, 0.2])
hiax.axis('off')

# Plot Hawaii in the Hawaii axis
gdf[gdf.STUSPS == 'HI'].plot(column='APNCU Score Numerical', cmap='pink', linewidth=0

# Adjust Hawaii position
hiax.set_position([0.15, 0.12, 0.2, 0.29])

# Set the title outside of the boxes with increased font size
fig.suptitle('Adequacy of Prenatal Healthcare Utilization in the United States\n (Jan

cmap = plt.get_cmap('pink')

# Generate colors from the colormap based on the numerical values
category_colors = {category: cmap(value/100) for category, value in category_mapping.

# Create a legend with colored dots using the generated colors
legend_handles = []
for category, color in category_colors.items():
    legend_handles = [mpatches.Patch(color=cmap(i/len(category_mapping)), label=categ
```



```

hiax.set_position([0.15, 0.12, 0.2, 0.29])

# Set the title outside of the boxes with increased font size
fig.suptitle('Average Month Prenatal Care Began in the United States\n (January 2023

cmap = plt.get_cmap('pink')

# Generate colors from the colormap based on the numerical values
category_colors = {category: cmap(value/100) for category, value in category_mapping.

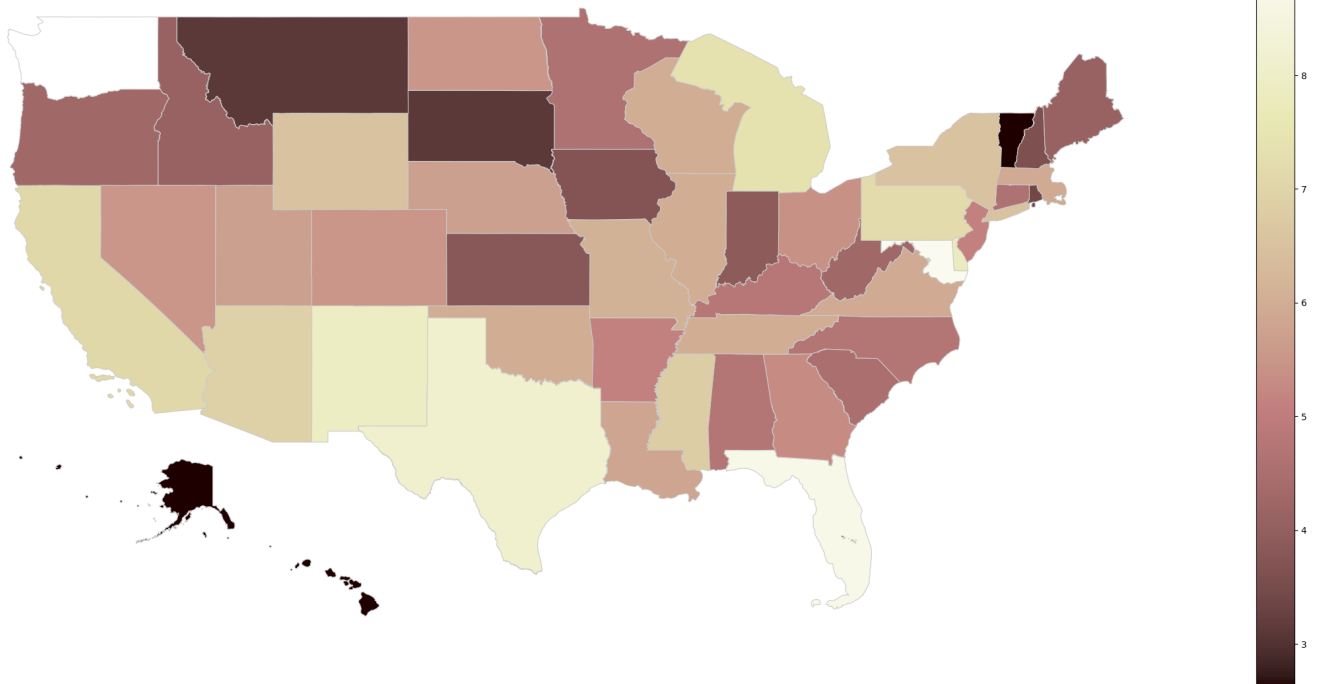
# Create the plot
fig, ax = plt.subplots(figsize=(18, 14))
ax.axis('off')

plt.subplots_adjust(bottom=0.8)
plt.savefig('prenatal_began_of_mothers_by_state.png', transparent=True)

# Plot the map of the United States
plt.show()

```

Average Month Prenatal Care Began in the United States (January 2023 - February 2024)



```

In [309... fig, ax = plt.subplots(figsize=(30, 15))
ax.axis('off')# Remove the axis box from the main map

# Plot all states except Alaska and Hawaii in the main map axis with APNCU Score Nume
gdf[~gdf.STUSPS.isin(['HI', 'AK'])].plot(column='Average Number of Prenatal Visits',

# Add Alaska Axis (x, y, width, height) – Increase width and height
akax = fig.add_axes([0.2, 0.2, 0.4, 0.2])
akax.axis('off')

# Plot Alaska in the Alaska axis
gdf[gdf.STUSPS == 'AK'].plot(column='Average Number of Prenatal Visits', cmap='pink',

# Add Hawaii Axis (x, y, width, height) – Increase width and height
hiax = fig.add_axes([0.28, 0.22, 0.2, 0.2])
hiax.axis('off')

```

```

# Plot Hawaii in the Hawaii axis
gdf[gdf.STUSPS == 'HI'].plot(column='Average Number of Prenatal Visits', cmap='pink',

# Adjust Hawaii position
hiax.set_position([0.15, 0.12, 0.2, 0.29])

# Set the title outside of the boxes with increased font size
fig.suptitle('Average Number of Prenatal Care Visits in the United States\n (January

cmap = plt.get_cmap('pink')

# Generate colors from the colormap based on the numerical values
category_colors = {category: cmap(value/100) for category, value in category_mapping.

# Create the plot
fig, ax = plt.subplots(figsize=(18, 14))
ax.axis('off')

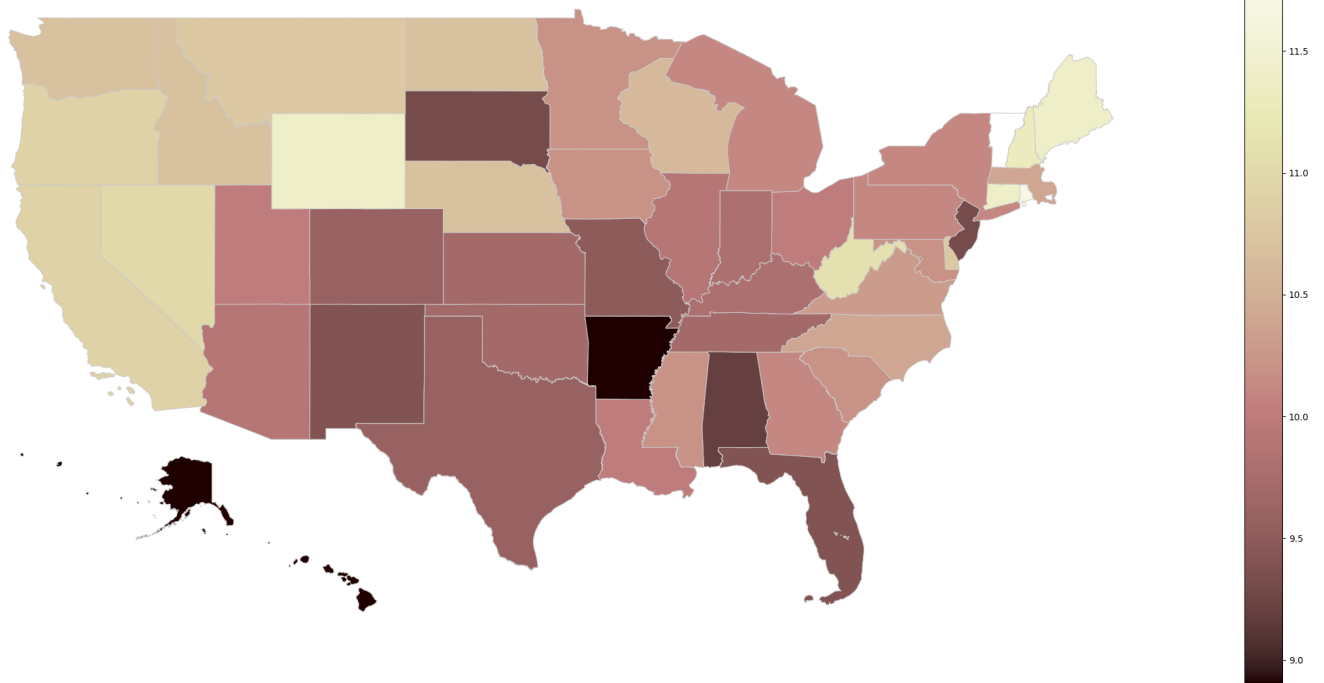
plt.subplots_adjust(bottom=0.8)
# Plot the map of the United States

plt.savefig('prenatal_visits_of_mothers_by_state.png', transparent=True)

plt.show()

```

Average Number of Prenatal Care Visits in the United States (January 2023 - February 2024)



```
In [267... new_raw_data = pd.read_excel('/Users/keriwilkins/Library/CloudStorage/OneDrive-Person
```

```
In [268... print(new_raw_data)
```

	State of Residence	Maternal Morbidity Checked	\
0	Alabama	At least one checked	
1	Alabama	None checked	
2	Alabama	None checked	
3	Alabama	None checked	
4	Alabama	None checked	
...	...	...	
22472	NaN	NaN	
22473	NaN	NaN	
22474	NaN	NaN	
22475	NaN	NaN	
22476	NaN	NaN	

	Maternal Morbidity Checked	Code	Source of Payment for Delivery
0	0.0		Medicaid
1	1.0		Private Insurance
2	1.0		Self Pay
3	1.0		Other
4	1.0		Unknown or Not Stated
...	...	...	...
22472	NaN		Medicaid
22473	NaN		Private Insurance
22474	NaN		Private Insurance
22475	NaN		Private Insurance
22476	NaN		Private Insurance

[22477 rows x 4 columns]

```
In [269... final_data = pd.merge(raw_data, new_raw_data, on='State of Residence', how='left')
```

```
In [270... print(final_data)
```

	State of Residence	State of Residence Code	Mother's Single Race 15	\
0	Alabama	1.0	White	
1	Alabama	1.0	White	
2	Alabama	1.0	White	
3	Alabama	1.0	White	
4	Alabama	1.0	White	
...	...	...	...	
13738092	Wyoming	56.0	More than one race	
13738093	Wyoming	56.0	More than one race	
13738094	Wyoming	56.0	More than one race	
13738095	Wyoming	56.0	More than one race	
13738096	Wyoming	56.0	More than one race	

	Mother's Single Race 15	Code	Number of Prenatal Visits Recode	\
0		1.0	No visits	
1		1.0	No visits	
2		1.0	No visits	
3		1.0	No visits	
4		1.0	No visits	
...		...	...	
13738092		15.0	11 to 12 visits	
13738093		15.0	11 to 12 visits	
13738094		15.0	11 to 12 visits	
13738095		15.0	11 to 12 visits	
13738096		15.0	11 to 12 visits	

	Number of Prenatal Visits Recode	Code	Month Prenatal Care Began	\
0		1.0	No prenatal care	
1		1.0	No prenatal care	
2		1.0	No prenatal care	
3		1.0	No prenatal care	
4		1.0	No prenatal care	
...		...	...	
13738092		7.0	2nd month	
13738093		7.0	2nd month	
13738094		7.0	2nd month	
13738095		7.0	2nd month	
13738096		7.0	2nd month	

	Month Prenatal Care Began	Code	OE Gestational Age Weekly	\
0		0.0	28 weeks	
1		0.0	28 weeks	
2		0.0	28 weeks	
3		0.0	28 weeks	
4		0.0	28 weeks	
...		...	...	
13738092		2.0	39 weeks	
13738093		2.0	39 weeks	
13738094		2.0	39 weeks	
13738095		2.0	39 weeks	
13738096		2.0	39 weeks	

	OE Gestational Age Weekly	Code	Births	Maternal Morbidity Checked	\
0		28.0	10.0	At least one checked	
1		28.0	10.0	None checked	
2		28.0	10.0	None checked	
3		28.0	10.0	None checked	
4		28.0	10.0	None checked	
...		...	...	...	
13738092		39.0	13.0	None checked	
13738093		39.0	13.0	None checked	
13738094		39.0	13.0	None checked	
13738095		39.0	13.0	None checked	
13738096		39.0	13.0	None checked	

	Maternal Morbidity Checked	Code	Source of Payment for Delivery
0	0.0		Medicaid
1	1.0		Private Insurance
2	1.0		Self Pay
3	1.0		Other
4	1.0		Unknown or Not Stated
...	...		...
13738092	1.0		Private Insurance
13738093	1.0		Other
13738094	1.0		Medicaid
13738095	1.0		Private Insurance
13738096	1.0		Self Pay

[13738097 rows x 14 columns]

```
In [271]: inadequate_states = APNCU_index_categories_df[APNCU_index_categories_df['APNCU Score'] < 2]
adequate_plus_states = APNCU_index_categories_df[APNCU_index_categories_df['APNCU Score'] >= 2]

# Print the list of states in each category
print("States with Inadequate Prenatal Care:")
print(inadequate_states.tolist())

print("\nStates with Adequate Plus Prenatal Care:")
print(adequate_plus_states.tolist())
```

States with Inadequate Prenatal Care:

['Arizona', 'California', 'Delaware', 'District of Columbia', 'Florida', 'Hawaii', 'Maryland', 'Michigan', 'Mississippi', 'New Mexico', 'New York', 'Pennsylvania', 'Texas', 'Washington', 'Wyoming']

States with Adequate Plus Prenatal Care:

['Alaska', 'Idaho', 'Indiana', 'Iowa', 'Kansas', 'Maine', 'Montana', 'New Hampshire', 'Oregon', 'Rhode Island', 'Vermont', 'West Virginia']

```
In [272]: # Filter the rows where 'Maternal Morbidity Checked' is 'At least one checked'
maternal_morbidity_checked = new_raw_data[new_raw_data['Maternal Morbidity Checked'] > 0]

# Group by 'State of Residence' and count the occurrences
maternal_morbidity_counts = maternal_morbidity_checked.groupby('State of Residence').count()

# Merge with the population data
maternal_morbidity_counts = maternal_morbidity_counts.merge(population_data, on='State of Residence')

# Calculate normalized count per 1000 births
maternal_morbidity_counts['Morbidity Count per 1000 Births'] = maternal_morbidity_counts['Morbidity Count per 1000 Births'] / maternal_morbidity_counts['Population']

# Merge with the original DataFrame to add the normalized count column
normalized_raw_data = normalized_raw_data.merge(maternal_morbidity_counts[['State of Residence', 'Morbidity Count per 1000 Births']], on='State of Residence', how='left')

# Fill missing values with 0
normalized_raw_data['Morbidity Count per 1000 Births'] = normalized_raw_data['Morbidity Count per 1000 Births'].fillna(0)
```

```
In [273]: # Filter maternal morbidity checked states from final_data based on inadequate and adequate plus categories
maternal_morbidity_inadequate = final_data[final_data['State of Residence'].isin(inadequate_states)]
maternal_morbidity_adequate_plus = final_data[final_data['State of Residence'].isin(adequate_plus_states)]

# Count occurrences of each state for both inadequate and adequate plus categories
inadequate_state_counts = maternal_morbidity_inadequate['State of Residence'].value_counts()
adequate_plus_state_counts = maternal_morbidity_adequate_plus['State of Residence'].value_counts()

# Get the list of states
inadequate_states_list = inadequate_state_counts.index.tolist()
adequate_plus_states_list = adequate_plus_state_counts.index.tolist()

# Get the counts of maternal morbidity for each state
```

```

adequate_plus_counts = [inadequate_state_counts[state] if state in inadequate_state_counts
adequate_plus_counts = [adequate_plus_state_counts[state] if state in adequate_plus_s

```

```

In [274... maternal_morbidity_check = new_raw_data[new_raw_data['Maternal Morbidity Checked'] ==

# Group by 'State of Residence' and count the occurrences
maternal_morbidity_counts = maternal_morbidity_checked.groupby('State of Residence').

# Merge with the original DataFrame to add the normalized count column
final_data = final_data.merge(maternal_morbidity_counts, on='State of Residence', how

# Fill missing values with 0
final_data['Maternal Morbidity Count'] = final_data['Maternal Morbidity Count'].fillna

```

```

In [320... def get_label_rotation(angle, offset):
    # Rotation must be specified in degrees :(
    rotation = np.rad2deg(angle + offset)
    if angle <= np.pi:
        alignment = "right"
        rotation = rotation + 180
    else:
        alignment = "left"
    return rotation, alignment

ANGLES = np.linspace(0, 2 * np.pi, len(normalized_raw_data), endpoint=False)
VALUES = normalized_raw_data['Morbidity Count per 1000 Births'].values * 100 # Increase
LABELS = normalized_raw_data['State of Residence'].values
# Convert Morbidity Count Per 1000 Births to percentages
percentages = VALUES * 100

# Determine the width of each bar.
# The circumference is '2 * pi', so we divide that total width over the number of bars
WIDTH = 2 * np.pi / len(VALUES)

# Determines where to place the first bar.
# By default, matplotlib starts at 0 (the first bar is horizontal)
# but here we say we want to start at pi/2 (90 deg)
OFFSET = np.pi / 2

# Initialize Figure and Axis
fig, ax = plt.subplots(figsize=(45, 38), subplot_kw={"projection": "polar"})

# Specify offset
ax.set_theta_offset(OFFSET)

# Set limits for radial (y) axis. The negative lower bound creates the whole in the middle
ax.set_ylim(-100, 100)

# Remove all spines
ax.set_frame_on(False)

# Remove grid and tick marks
ax.xaxis.grid(False)
ax.yaxis.grid(False)
ax.set_xticks([])
ax.set_yticks([])

# Add labels
def add_labels(angles, values, labels, offset, ax):

    # This is the space between the end of the bar and the label
    padding = 10 # Change this value for different padding

    # Iterate over angles, values, and labels, to add all of them.
    for angle, value, label, in zip(angles, values, labels):

```

```

angle = angle

# Obtain text rotation and alignment
rotation, alignment = get_label_rotation(angle, offset)

# And finally add the text
ax.text(
    x=angle,
    y=value + padding,
    s=label,
    ha=alignment,
    va="center",
    rotation=rotation,
    rotation_mode="anchor",
    fontsize=45,
    fontweight='bold'
)

# Add labels
add_labels(ANGLES, percentages, LABELS, OFFSET, ax)

# Add bars with colormap
bars = ax.bar(
    ANGLES, percentages, width=WIDTH, linewidth=2,
    color=plt.cm.pink_r(percentages / 100), edgecolor="white"
)

# Add percentages as text inside the circle
percentage_padding = -50 # Adjust this value for the desired padding
for angle, percentage in zip(ANGLES, percentages):
    ax.text(
        x=angle,
        y=percentage + percentage_padding,
        s=f'{percentage:.1f}%',
        ha='center',
        va='center',
        fontsize=45,
        fontweight='bold',
        color='black'
    )

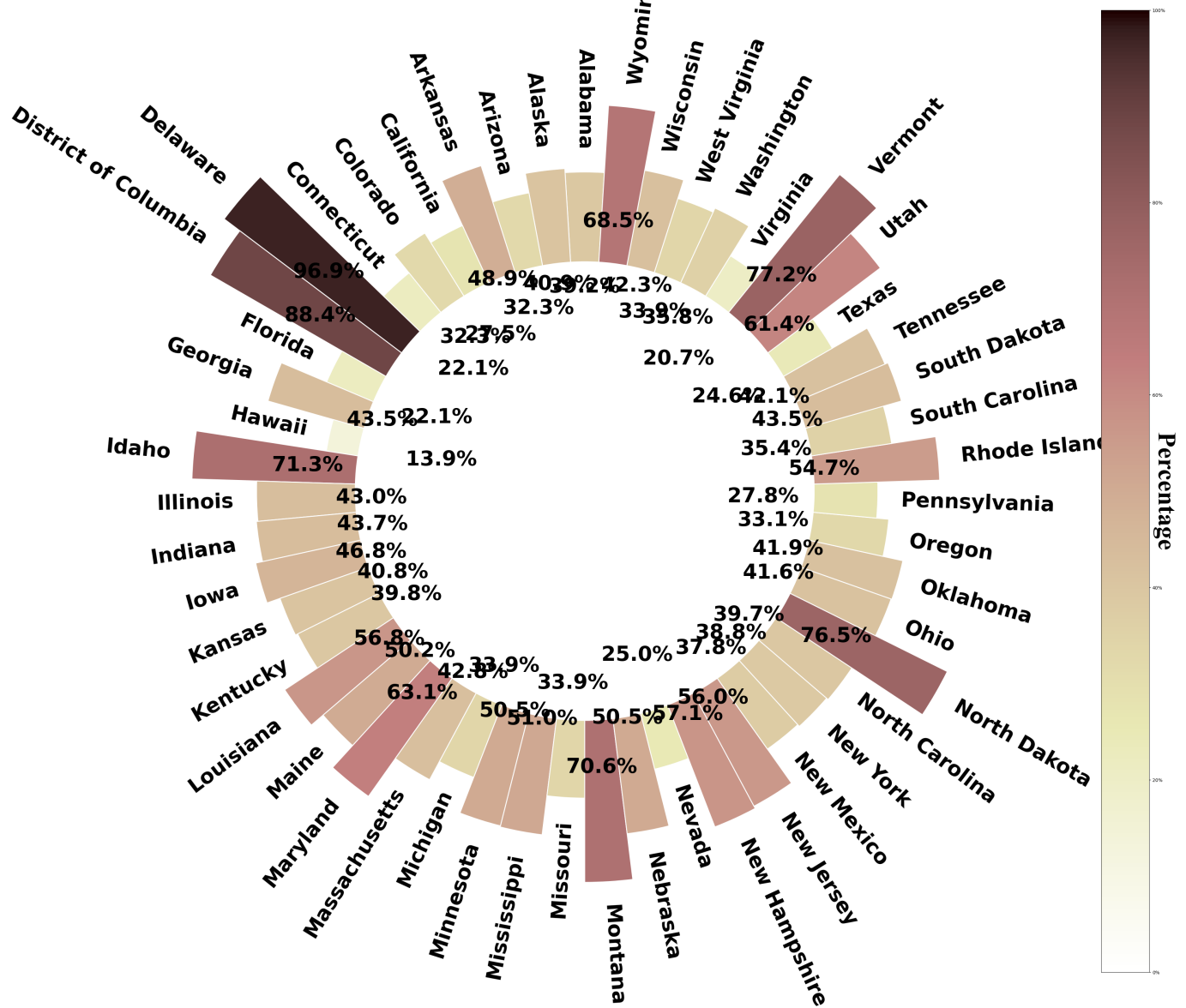
# Add colorbar legend
cbar = plt.colorbar(plt.cm.ScalarMappable(cmap='pink_r', norm=plt.Normalize(vmin=0, v
cbar.set_label('Percentage', rotation=270, labelpad=20, fontsize=55, fontweight='bold

plt.title('Maternal Morbidity Count Per 1000 Births By State \n (Percentage)', fontsi
plt.savefig('maternal_morbidity_of_mothers_by_state.png', transparent=True)

plt.show()

```

Maternal Morbidity Count Per 1000 Births By State
(Percentage)



```
In [276... # Filter final_data to include only records where Maternal Morbidity Checked is 'At least'
filtered_final_data = final_data[final_data['Maternal Morbidity Checked'] == 'At least']

# Count the occurrences of each unique payment source by state
payment_source_counts = filtered_final_data.groupby(['State of Residence', 'Source of Payment']).count()

# Merge with normalized_raw_data to get population data
payment_source_data = pd.merge(payment_source_counts, normalized_raw_data[['State of Residence', 'Population']], on='State of Residence')

for column in payment_source_counts.columns:
    payment_source_data[column] = payment_source_data[column] / payment_source_data['Population']

# Display the result
print(payment_source_data)
```

	State of Residence	Medicaid	Other	Private Insurance	Self Pay \
0	Alabama	0.062347	0.074817	0.099756	0.012469
1	Alaska	0.028497	0.056994	0.000000	0.000000
2	Arizona	0.095380	0.059612	0.083457	0.035767
3	Arkansas	0.046647	0.046647	0.077745	0.062196
4	California	0.272931	0.055827	0.179886	0.124059
5	Colorado	0.057166	0.057166	0.045733	0.034300
6	Connecticut	0.024826	0.024826	0.037239	0.012413
7	Delaware	0.044385	0.088769	0.066577	0.022192
8	District of Columbia	0.029162	0.029162	0.087485	0.029162
9	Florida	0.149062	0.027102	0.149062	0.013551
10	Georgia	0.110343	0.143446	0.165515	0.099309
11	Hawaii	0.024945	0.000000	0.024945	0.000000
12	Idaho	0.051712	0.012928	0.064640	0.051712
13	Illinois	0.151446	0.062360	0.106903	0.151446
14	Indiana	0.089417	0.122949	0.100595	0.022354
15	Iowa	0.077081	0.000000	0.051388	0.051388
16	Kansas	0.050739	0.025369	0.025369	0.050739
17	Kentucky	0.063962	0.012792	0.115131	0.038377
18	Louisiana	0.151517	0.030303	0.136365	0.060607
19	Maine	0.038475	0.000000	0.025650	0.025650
20	Maryland	0.154314	0.084171	0.196400	0.084171
21	Massachusetts	0.070086	0.050061	0.090111	0.070086
22	Michigan	0.156537	0.039134	0.097835	0.029351
23	Minnesota	0.098154	0.036808	0.134962	0.085885
24	Mississippi	0.085893	0.051536	0.068715	0.051536
25	Missouri	0.081115	0.046351	0.104290	0.000000
26	Montana	0.080772	0.016154	0.000000	0.032309
27	Nebraska	0.050344	0.016781	0.050344	0.033563
28	Nevada	0.032497	0.000000	0.048745	0.048745
29	New Hampshire	0.023965	0.000000	0.035947	0.023965
30	New Jersey	0.204298	0.032258	0.204298	0.107525
31	New Mexico	0.050653	0.016884	0.016884	0.050653
32	New York	0.319009	0.067160	0.167900	0.050370
33	North Carolina	0.132786	0.088524	0.188113	0.066393
34	North Dakota	0.061996	0.020665	0.041330	0.000000
35	Ohio	0.120228	0.055490	0.166470	0.073986
36	Oklahoma	0.123340	0.030835	0.061670	0.030835
37	Oregon	0.047244	0.011811	0.059055	0.047244
38	Pennsylvania	0.125292	0.044747	0.089495	0.053697
39	Rhode Island	0.015968	0.000000	0.063871	0.015968
40	South Carolina	0.065227	0.065227	0.026091	0.091318
41	South Dakota	0.021646	0.021646	0.021646	0.021646
42	Tennessee	0.103557	0.046025	0.138076	0.057532
43	Texas	0.164913	0.096199	0.116814	0.109942
44	Utah	0.096906	0.048453	0.096906	0.012113
45	Vermont	0.025330	0.025330	0.012665	0.000000
46	Virginia	0.056737	0.000000	0.045389	0.045389
47	Washington	0.066301	0.013260	0.092821	0.079561
48	West Virginia	0.041863	0.000000	0.013954	0.013954
49	Wisconsin	0.154899	0.000000	0.077449	0.044257
50	Wyoming	0.018834	0.018834	0.037668	0.000000

	Unknown or Not Stated	2023
0	0.000000	5108468
1	0.000000	733406
2	0.011922	7431344
3	0.000000	3067732
4	0.031015	38965193
5	0.022866	5877610
6	0.000000	3617176
7	0.000000	1031890
8	0.000000	678972
9	0.000000	22610726
10	0.011034	11029227

11	0.000000	1435138
12	0.000000	1964726
13	0.008909	12549689
14	0.000000	6862199
15	0.012847	3207004
16	0.000000	2940546
17	0.000000	4526154
18	0.015152	4573749
19	0.000000	1395722
20	0.028057	6180253
21	0.020025	7001399
22	0.009784	10037261
23	0.000000	5737915
24	0.000000	2939690
25	0.011588	6196156
26	0.000000	1132812
27	0.016781	1978379
28	0.000000	3194176
29	0.011982	1402054
30	0.010753	9290841
31	0.000000	2114371
32	0.033580	19571216
33	0.000000	10835491
34	0.000000	783926
35	0.036993	11785935
36	0.015418	4053824
37	0.000000	4233358
38	0.008949	12961683
39	0.000000	1095962
40	0.000000	5373555
41	0.000000	919318
42	0.000000	7126489
43	0.027486	30503301
44	0.000000	3417734
45	0.000000	647464
46	0.056737	8715698
47	0.119341	7812880
48	0.013954	1770071
49	0.000000	5910955
50	0.000000	584057

```
In [295... # Get the payment sources and their percentages for Delaware
percentages = payment_source_data[payment_source_data['State of Residence'] == 'Delaw

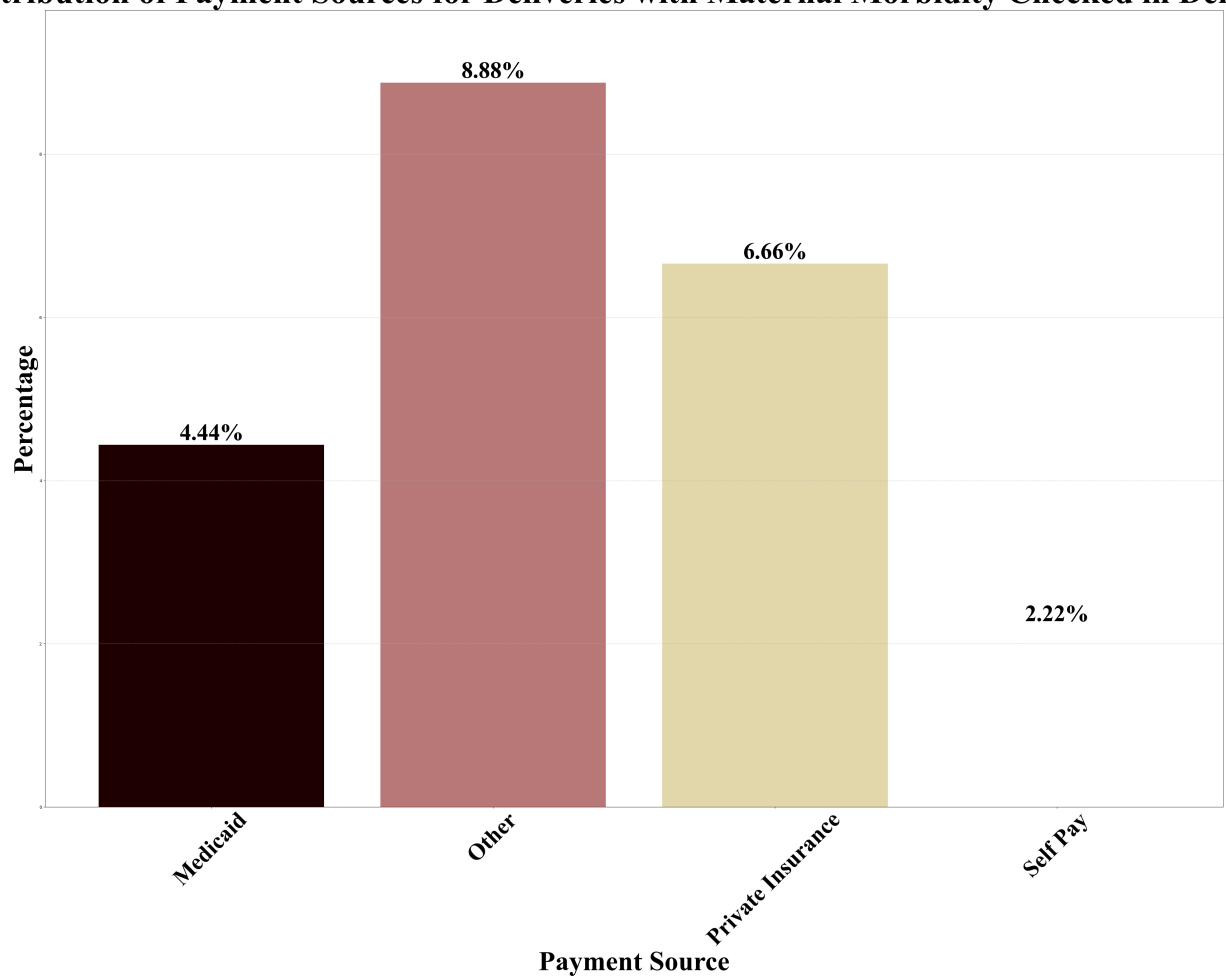
# Define colors using a colormap
colors = plt.cm.pink(np.linspace(0, 1, len(percentages)))

# Plotting
plt.figure(figsize=(40, 33))
bars = plt.bar(['Medicaid', 'Other', 'Private Insurance', 'Self Pay'], percentages, c
plt.title('Distribution of Payment Sources for Deliveries with Maternal Morbidity Che
plt.xlabel('Payment Source', fontsize=65, fontweight='bold', fontname='Times New Roma
plt.ylabel('Percentage', fontsize=65, fontweight='bold', fontname='Times New Roman')
plt.ylim(0, max(percentages) * 1.1) # Setting y-axis limit to provide some padding
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.xticks(rotation=45, fontsize=55, fontweight='bold', fontname='Times New Roman') #

for bar, percentage in zip(bars, percentages):
    height = bar.get_height()
    plt.text(bar.get_x() + bar.get_width() / 2, height, f'{percentage:.2f}%', ha='cen

plt.tight_layout()
plt.savefig('payment_delaware.png', transparent=True)
plt.show()
```

Distribution of Payment Sources for Deliveries with Maternal Morbidity Checked in Delaware



```
In [300... # Get the payment sources and their percentages for Delaware
percentages = payment_source_data[payment_source_data['State of Residence'] == 'Hawaii']

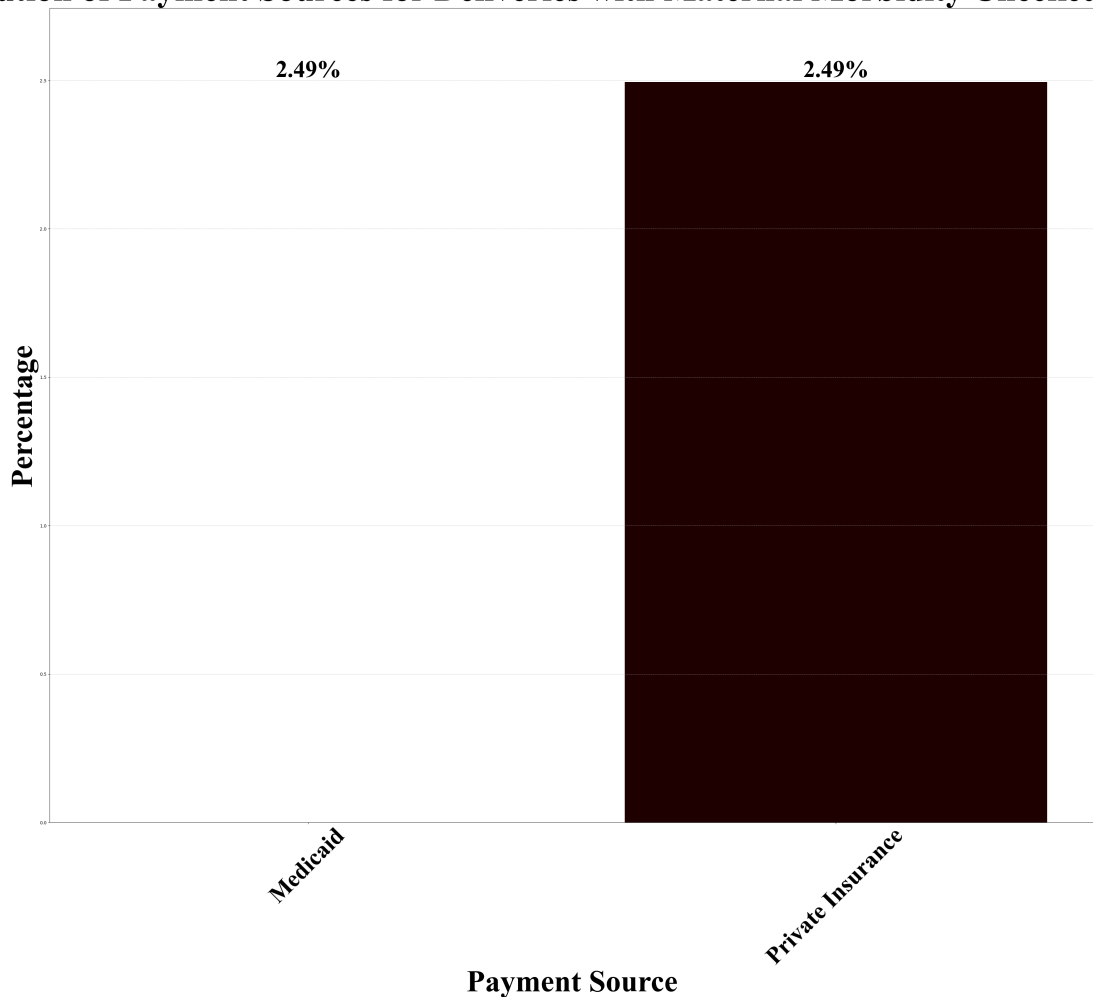
# Define colors using a colormap
colors = plt.cm.pink_r(np.linspace(0, 1, len(percentages)))

# Plotting
plt.figure(figsize=(35, 33))
bars = plt.bar(['Medicaid', 'Private Insurance'], percentages, color=colors)
plt.title('Distribution of Payment Sources for Deliveries with Maternal Morbidity Checked in Delaware')
plt.xlabel('Payment Source', fontsize=70, fontweight='bold', fontname='Times New Roman')
plt.ylabel('Percentage', fontsize=70, fontweight='bold', fontname='Times New Roman')
plt.ylim(0, max(percentages) * 1.1) # Setting y-axis limit to provide some padding
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.xticks(rotation=45, fontsize=55, fontweight='bold', fontname='Times New Roman') #

for bar, percentage in zip(bars, percentages):
    height = bar.get_height()
    plt.text(bar.get_x() + bar.get_width() / 2, height, f'{percentage:.2f}%', ha='center')

plt.tight_layout()
plt.savefig('payment_hawaii.png', transparent=True)
plt.show()
```

Distribution of Payment Sources for Deliveries with Maternal Morbidity Checked in Hawaii



```
In [306... # Get the payment sources and their percentages for Delaware
percentages = payment_source_data[payment_source_data['State of Residence'] == 'Distr

# Define colors using a colormap
colors = plt.cm.pink(np.linspace(0, 1, len(percentages)))

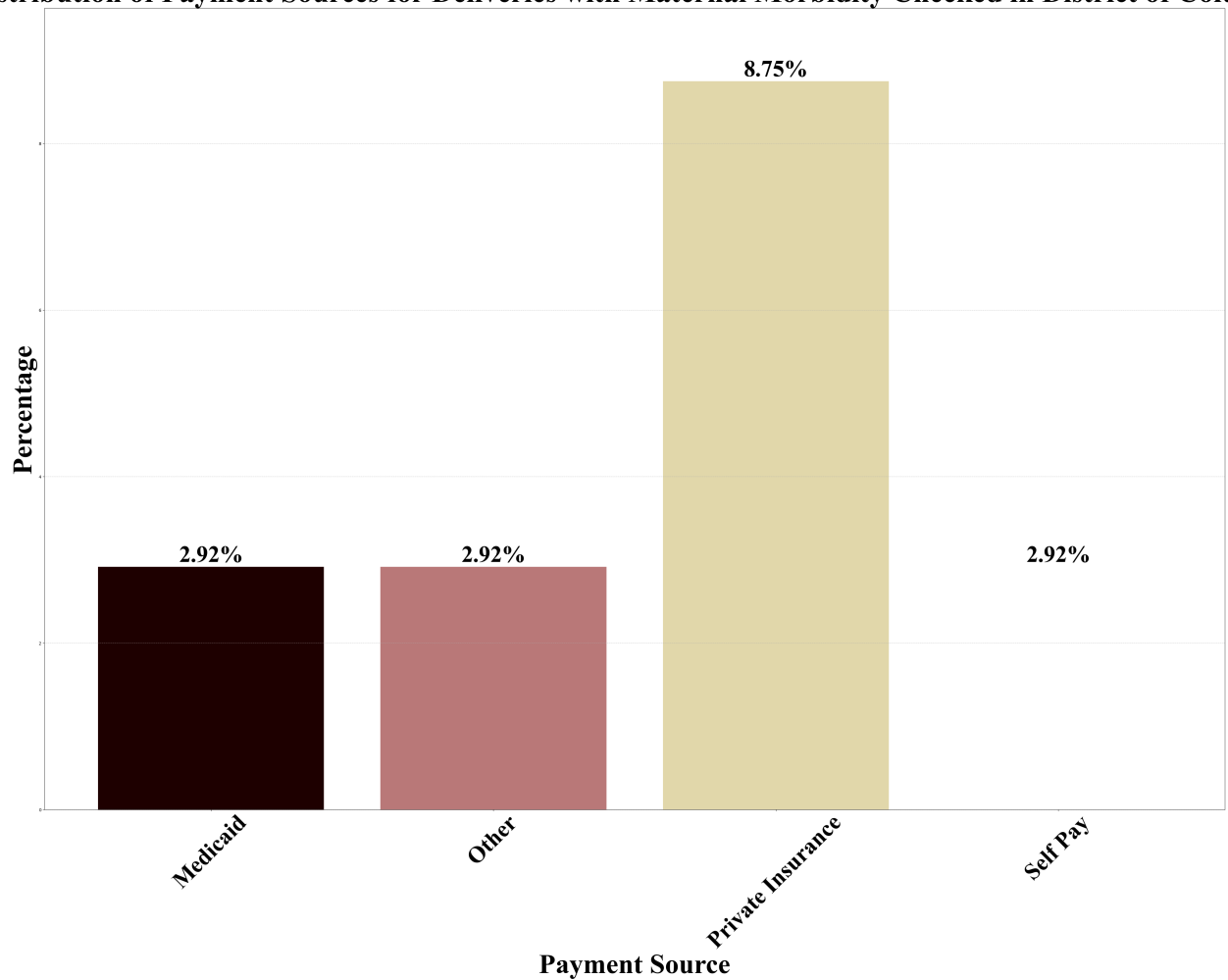
# Plotting
plt.figure(figsize=(40, 33))
bars = plt.bar(['Medicaid', 'Other', 'Private Insurance', 'Self Pay'], percentages, c
plt.title('Distribution of Payment Sources for Deliveries with Maternal Morbidity Che
plt.xlabel('Payment Source', fontsize=65, fontweight='bold', fontname='Times New Roma
plt.ylabel('Percentage', fontsize=65, fontweight='bold', fontname='Times New Roman')
plt.ylim(0, max(percentages) * 1.1) # Setting y-axis limit to provide some padding
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.xticks(rotation=45, fontsize=55, fontweight='bold', fontname='Times New Roman') #

for bar, percentage in zip(bars, percentages):
    height = bar.get_height()
    plt.text(bar.get_x() + bar.get_width() / 2, height, f'{percentage:.2f}%', ha='cen

plt.tight_layout()
plt.savefig('payment_dc.png', transparent=True)

plt.show()
```

Distribution of Payment Sources for Deliveries with Maternal Morbidity Checked in District of Columbia



```
In [305... # Get the payment sources and their percentages for Delaware
percentages = payment_source_data[payment_source_data['State of Residence'] == 'Flori

# Define colors using a colormap
colors = plt.cm.pink(np.linspace(0, 1, len(percentages)))

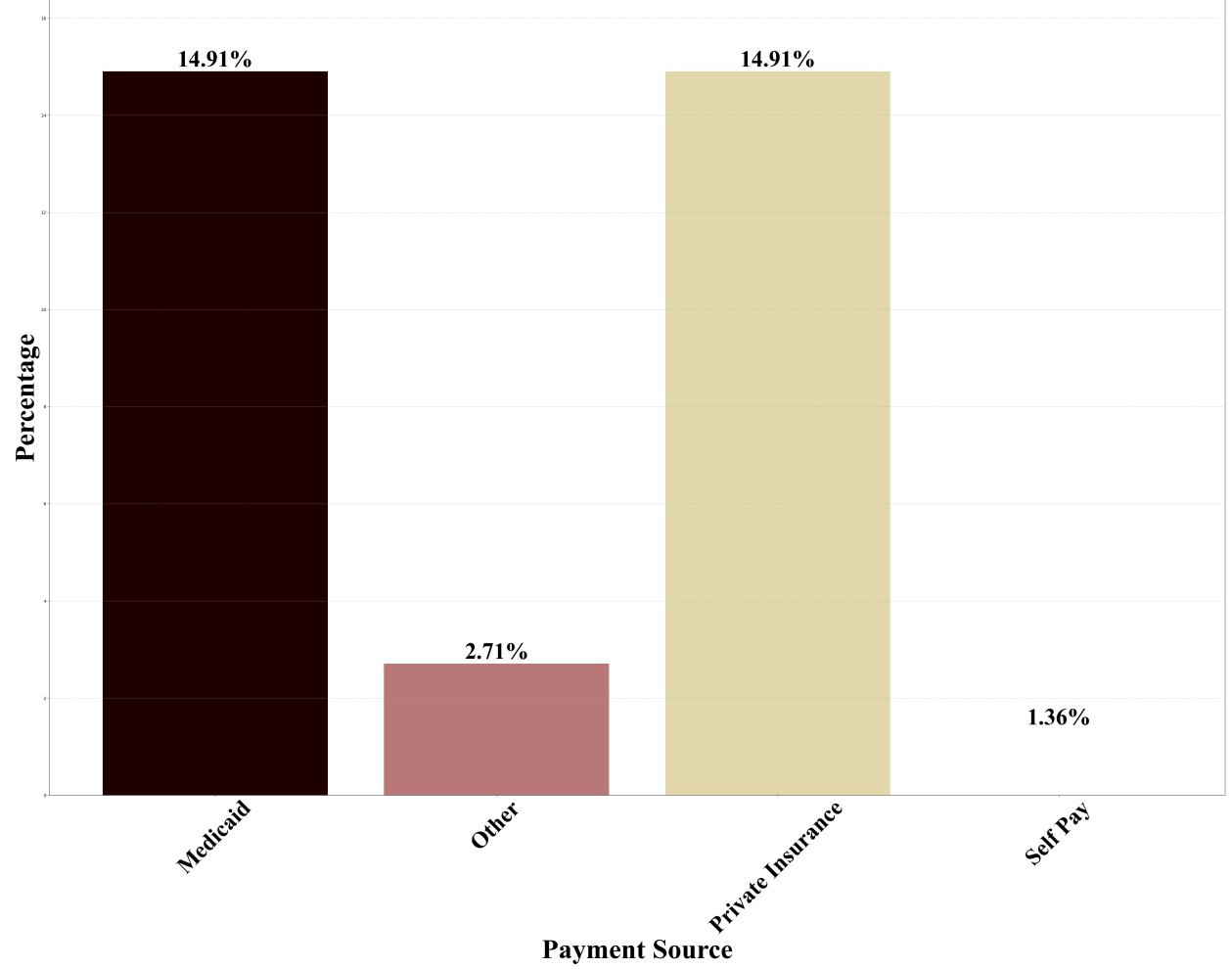
# Plotting
plt.figure(figsize=(40, 33))
bars = plt.bar(['Medicaid', 'Other', 'Private Insurance', 'Self Pay'], percentages, c
plt.title('Distribution of Payment Sources for Deliveries with Maternal Morbidity Che
plt.xlabel('Payment Source', fontsize=65, fontweight='bold', fontname='Times New Roma
plt.ylabel('Percentage', fontsize=65, fontweight='bold', fontname='Times New Roman')
plt.ylim(0, max(percentages) * 1.1) # Setting y-axis limit to provide some padding
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.xticks(rotation=45, fontsize=55, fontweight='bold', fontname='Times New Roman') #

for bar, percentage in zip(bars, percentages):
    height = bar.get_height()
    plt.text(bar.get_x() + bar.get_width() / 2, height, f'{percentage:.2f}%', ha='cen

plt.tight_layout()
plt.savefig('payment_fl.png', transparent=True)

plt.show()
```

Distribution of Payment Sources for Deliveries with Maternal Morbidity Checked in Florida



In []: