Merge All Data Needed for ArcGIS Maps

1. State Level Data

Disease Prevalence Dataframe

In [ ]:

```python
import pandas as pd
# Health variables
df = pd.read_csv("C:/Users/user/Desktop/GIS/Final Project/PLACES__County_Data__GIS_Friendly_Format___2023_release_20240506.csv")
df.head(3)
```

Out[ ]:

	StateAbbr	StateDesc	CountyName	CountyFIPS	TotalPopulation	ACCESS2_CrudePrev	ACCESS2_Crude95CI	ACCESS2_AdjPrev	ACCESS2_Adj95CI	ARTHR
0	AL	Alabama	Autauga	1001	59095	10.0	(7.7, 12.6)	10.4	(8.0, 13.1)	
1	AL	Alabama	Bullock	1011	10320	18.7	(15.3, 23.0)	19.2	(15.7, 23.6)	
2	AL	Alabama	Chilton	1021	45274	13.5	(10.8, 16.9)	14.1	(11.2, 17.6)	

3 rows × 154 columns

In [ ]:

```python
# Filter columns that end with '_CrudePrev'
columns_to_group =[col for col in df.columns if col.endswith('_AdjPrev')]

# Group by StateAbbr and aggregate the columns using mean()
state_df = df.groupby(['StateAbbr', "StateDesc"])[columns_to_group].mean().reset_index()
state_df.head(3)
```

Out[ ]:

	StateAbbr	StateDesc	ACCESS2_AdjPrev	ARTHRITIS_AdjPrev	BINGE_AdjPrev	BPHIGH_AdjPrev	BPMED_AdjPrev	CANCER_AdjPrev	CASTHMA_AdjPrev	C
0	AK	Alaska	12.900000	22.083333	18.556667	30.243333	51.640000	6.076667	10.126667	
1	AL	Alabama	13.220896	28.986567	14.617910	39.455224	66.020896	6.153731	10.883582	
2	AR	Arkansas	11.934667	27.272000	14.470667	38.052000	62.222667	6.208000	10.177333	

3 rows × 39 columns

In [ ]:

```python
state_df_filtered = state_df[['StateAbbr', "StateDesc", 'COPD_AdjPrev', "ACCESS2_AdjPrev", 'BINGE_AdjPrev', 'CHECKUP_AdjPrev', 'CS
                              'LPA_AdjPrev', 'SLEEP_AdjPrev']]
```

Add Unemployment Dataframe

In [ ]:

```python
econ_df = pd.read_excel("C:/Users/user/Desktop/GIS/Final Project/Unemployment.xlsx")
econ_df.head(3)
```

Out[ ]:

	FIPS_Code	State	Area_Name	Rural_Urban_Continuum_Code_2013	Urban_Influence_Code_2013	Metro_2013	Civilian_labor_force_2000	Employed_2000	U
0	0	US	United States	NaN	NaN	NaN	142601576.0	136904853.0	
1	1000	AL	Alabama	NaN	NaN	NaN	2147173.0	2047731.0	
2	1001	AL	Autauga County, AL	2.0	2.0	1.0	21861.0	20971.0	

3 rows × 100 columns

In [ ]:

```python
econ_df = econ_df[['State','Unemployment_rate_2021',"Median_Household_Income_2021"]]
# Group by StateAbbr and aggregate the columns using mean()
econ_df = econ_df.groupby('State').mean().reset_index()
```

In [ ]:

```python
# Merge aggregated DataFrame with the GeoDataFrame
merged_df = state_df_filtered.merge(econ_df, left_on='StateAbbr', right_on='State', how='left')
merged_df.head(3)
```

Out[ ]:

	StateAbbr	StateDesc	COPD_AdjPrev	ACCESS2_AdjPrev	BINGE_AdjPrev	CHECKUP_AdjPrev	CSMOKING_AdjPrev	LPA_AdjPrev	SLEEP_AdjPrev	State	Un
0	AK	Alaska	6.996667	12.900000	18.556667	61.393333	21.666667	24.240000	32.700000	AK	
1	AL	Alabama	8.208955	13.220896	14.617910	75.746269	20.647761	34.652239	40.441791	AL	
2	AR	Arkansas	8.493333	11.934667	14.470667	75.737333	23.032000	33.020000	35.620000	AR	

Add Poverty Dataframe

In [ ]:
```
poverty = pd.read_excel("C:/Users/user/Desktop/GIS/Final Project/PovertyEstimates.xlsx")
poverty.head(3)
```

Out[ ]:

	FIPS_Code	Stabr	Area_name	Rural-urban_Continuum_Code_2003	Urban_Influence_Code_2003	Rural-urban_Continuum_Code_2013	Urban_Influence_Code_2013	POVALL_2
0	0	US	United States	NaN	NaN	NaN	NaN	413931
1	1000	AL	Alabama	NaN	NaN	NaN	NaN	8008
2	1001	AL	Autauga County	2.0	2.0	2.0	2.0	62

3 rows × 34 columns

In [ ]:
```
poverty = poverty[['Stabr','PCTPOVALL_2021']]
poverty = poverty.groupby('Stabr').mean().reset_index()
```

In [ ]:
```
# Merge aggregated DataFrame with the GeoDataFrame
merged_df = merged_df.merge(poverty, left_on='StateAbbr', right_on='Stabr', how='left')
merged_df.head(3)
```

Out[ ]:

	StateAbbr	StateDesc	COPD_AdjPrev	ACCESS2_AdjPrev	BINGE_AdjPrev	CHECKUP_AdjPrev	CSMOKING_AdjPrev	LPA_AdjPrev	SLEEP_AdjPrev	State	Un
0	AK	Alaska	6.996667	12.900000	18.556667	61.393333	21.666667	24.240000	32.700000	AK	
1	AL	Alabama	8.208955	13.220896	14.617910	75.746269	20.647761	34.652239	40.441791	AL	
2	AR	Arkansas	8.493333	11.934667	14.470667	75.737333	23.032000	33.020000	35.620000	AR	

Add Education Dataframe

In [ ]:
```
educ_df = pd.read_excel("C:/Users/user/Desktop/GIS/Final Project/Education.xlsx")
educ_df.head(3)
```

Out[ ]:

	Federal Information Processing Standard (FIPS) Code	State	Area name	2003 Rural-urban Continuum Code	2003 Urban Influence Code	2013 Rural-urban Continuum Code	2013 Urban Influence Code	Less than a high school diploma, 1970	High school diploma only, 1970	Some college (1-3 years), 1970	...	Percent of adults completing some college or associate's degree, 2008-12	Percent of adults with a bachelor's degree or higher, 2008-12	Less than a high school diploma, 2017-21
0	0	US	United States	NaN	NaN	NaN	NaN	52373312.0	34158051.0	11650730.0	...	28.993579	28.484955	25050356.0
1	1000	AL	Alabama	NaN	NaN	NaN	NaN	1062306.0	468269.0	136287.0	...	29.022866	22.264896	430047.0
2	1001	AL	Autauga County	2.0	2.0	2.0	2.0	6611.0	3757.0	933.0	...	29.618142	21.707831	4126.0

3 rows × 55 columns

In [ ]:
```
educ_df = educ_df[['State',"Percent of adults with a bachelor's degree or higher, 2017-21"]]
educ_df = educ_df.groupby('State').mean().reset_index()
```

In [ ]:
```
# Merge aggregated DataFrame with the GeoDataFrame
merged_df = merged_df.merge(educ_df, left_on='StateAbbr', right_on='State', how='left')
merged_df.head(3)
```

Out[ ]:

	StateAbbr	StateDesc	COPD_AdjPrev	ACCESS2_AdjPrev	BINGE_AdjPrev	CHECKUP_AdjPrev	CSMOKING_AdjPrev	LPA_AdjPrev	SLEEP_AdjPrev	State_x
0	AK	Alaska	6.996667	12.900000	18.556667	61.393333	21.666667	24.240000	32.700000	AK
1	AL	Alabama	8.208955	13.220896	14.617910	75.746269	20.647761	34.652239	40.441791	AL
2	AR	Arkansas	8.493333	11.934667	14.470667	75.737333	23.032000	33.020000	35.620000	AR

Add population Dataframe

In [ ]:

```
pop_df = pd.read_excel("C:/Users/user/Desktop/GIS/Final Project/PopulationEstimates.xlsx")
pop_df.head(3)
```

Out[ ]:

	FIPStxt	State	Area_Name	Rural_Urban_Continuum_Code_2003	Rural_Urban_Continuum_Code_2013	Urban_Influence_2003	Urban_Influence_2013	Econon
0	0	US	United States	NaN	NaN	NaN	NaN	
1	1000	AL	Alabama	NaN	NaN	NaN	NaN	
2	1001	AL	Autauga County	2.0	2.0	2.0	2.0	

3 rows × 53 columns

In [ ]:

```
pop_df = pop_df[['State','R_DEATH_2021',"R_BIRTH_2021",
                 "R_INTERNATIONAL_MIG_2021", "R_DOMESTIC_MIG_2021","R_NET_MIG_2021", "R_NATURAL_CHG_2021"]]
# Group by StateAbbr and aggregate the columns using mean()
pop_df = pop_df.groupby('State').mean().reset_index()
```

In [ ]:

```
# Merge aggregated DataFrame with the GeoDataFrame
merged_df = merged_df.merge(pop_df, left_on='StateAbbr', right_on='State', how='left')
merged_df.head(3)
```

Out[ ]:

	StateAbbr	StateDesc	COPD_AdjPrev	ACCESS2_AdjPrev	BINGE_AdjPrev	CHECKUP_AdjPrev	CSMOKING_AdjPrev	LPA_AdjPrev	SLEEP_AdjPrev	State_x
0	AK	Alaska	6.996667	12.900000	18.556667	61.393333	21.666667	24.240000	32.700000	AK
1	AL	Alabama	8.208955	13.220896	14.617910	75.746269	20.647761	34.652239	40.441791	AL
2	AR	Arkansas	8.493333	11.934667	14.470667	75.737333	23.032000	33.020000	35.620000	AR

3 rows × 23 columns

Add medical personnel Dataframe

In [ ]:

```
med_df = pd.read_csv("C:/Users/user/Desktop/GIS/Final Project/ahrfsn2023.csv")
med_df.head(3)
```

Out[ ]:

	fips_st	st_abbrev	phys_wkforc_21	phys_mal_21	phys_fem_21	phys_lt30_21	phys_30_39_21	phys_40_49_21	phys_50_59_21	phys_ge60_21	...	popn_ge
0	1	AL	11961	8473	3488	1191.0	2413.0	2282.0	2998.0	3077.0	...	
1	2	AK	2114	1216	898	NaN	815.0	476.0	NaN	NaN	...	
2	4	AZ	17347	11407	5940	976.0	4023.0	4674.0	4005.0	3669.0	...	

3 rows × 1432 columns

In [ ]:

```
med_df= med_df[['st_abbrev','phys_wkforc_21', "rn_21",
                "pharm_21", "socwk_21","pt_21", "resp_ther_21",
                "popn_pums_21", "popn_mal_21", "popn_50_59_21", "popn_ge60_21",
                "popn_wh_21", "popn_bl_21", "popn_hsp_21", "popn_asn_21"]]
```

In [ ]:

```
# Merge aggregated DataFrame with the GeoDataFrame
merged_df = merged_df.merge(med_df, left_on='StateAbbr', right_on='st_abbrev', how='left')
merged_df.head(3)
```

Out[ ]:

	StateAbbr	StateDesc	COPD_AdjPrev	ACCESS2_AdjPrev	BINGE_AdjPrev	CHECKUP_AdjPrev	CSMOKING_AdjPrev	LPA_AdjPrev	SLEEP_AdjPrev	State_x	.
0	AK	Alaska	6.996667	12.900000	18.556667	61.393333	21.666667	24.240000	32.700000	AK	.
1	AL	Alabama	8.208955	13.220896	14.617910	75.746269	20.647761	34.652239	40.441791	AL	.
2	AR	Arkansas	8.493333	11.934667	14.470667	75.737333	23.032000	33.020000	35.620000	AR	.

3 rows × 38 columns

Save State-Level Data to Excel File

In [ ]:

```
merged_df.to_excel("state_merged_data.xlsx")
```

2. County Level Data

Disease Prevalence Dataframe

In [ ]:

```
import pandas as pd
# Health variables
df = pd.read_csv("C:/Users/user/Desktop/GIS/Final Project/PLACES__County_Data__GIS_Friendly_Format___2023_release_20240506.csv")
df.head(3)
```

Out[ ]:

	StateAbbr	StateDesc	CountyName	CountyFIPS	TotalPopulation	ACCESS2_CrudePrev	ACCESS2_Crude95CI	ACCESS2_AdjPrev	ACCESS2_Adj95CI	ARTHR
0	AL	Alabama	Autauga	1001	59095	10.0	(7.7, 12.6)	10.4	(8.0, 13.1)	
1	AL	Alabama	Bullock	1011	10320	18.7	(15.3, 23.0)	19.2	(15.7, 23.6)	
2	AL	Alabama	Chilton	1021	45274	13.5	(10.8, 16.9)	14.1	(11.2, 17.6)	

3 rows × 154 columns

In [ ]:

```
df_filtered = df[['StateAbbr', "CountyName", "CountyFIPS", 'COPD_AdjPrev', "ACCESS2_AdjPrev", 'BINGE_AdjPrev', 'CHECKUP_AdjPrev',
                  'LPA_AdjPrev', 'SLEEP_AdjPrev']]
```

Add Unemployment Dataframe

In [ ]:

```
econ_df = pd.read_excel("C:/Users/user/Desktop/GIS/Final Project/Unemployment.xlsx")
econ_df.head(3)
```

Out[ ]:

	FIPS_Code	State	Area_Name	Rural_Urban_Continuum_Code_2013	Urban_Influence_Code_2013	Metro_2013	Civilian_labor_force_2000	Employed_2000	U
0	0	US	United States	NaN	NaN	NaN	142601576.0	136904853.0	
1	1000	AL	Alabama	NaN	NaN	NaN	2147173.0	2047731.0	
2	1001	AL	Autauga County, AL	2.0	2.0	1.0	21861.0	20971.0	

3 rows × 100 columns

In [ ]:

```
econ_df = econ_df[['State', "FIPS_Code", 'Unemployment_rate_2021',"Median_Household_Income_2021"]]
```

In [ ]:

```
# Merge aggregated DataFrame with the GeoDataFrame
merged_df = df_filtered.merge(econ_df, left_on='CountyFIPS', right_on='FIPS_Code', how='left')
merged_df.head(3)
```

Out[ ]:

	StateAbbr	CountyName	CountyFIPS	COPD_AdjPrev	ACCESS2_AdjPrev	BINGE_AdjPrev	CHECKUP_AdjPrev	CSMOKING_AdjPrev	LPA_AdjPrev	SLEEP_Ad
0	AL	Autauga	1001	6.8	10.4	15.5	76.0	16.9	29.1	
1	AL	Bullock	1011	9.8	19.2	12.4	78.2	25.7	43.6	
2	AL	Chilton	1021	8.3	14.1	15.8	72.7	21.7	34.1	

Add Poverty Dataframe

In [ ]:

```
poverty = pd.read_excel("C:/Users/user/Desktop/GIS/Final Project/PovertyEstimates.xlsx")
poverty.head(3)
```

Out[ ]:

	FIPS_Code	Stabr	Area_name	Rural-urban_Continuum_Code_2003	Urban_Influence_Code_2003	Rural-urban_Continuum_Code_2013	Urban_Influence_Code_2013	POVALL_2
0	0	US	United States	NaN	NaN	NaN	NaN	413931
1	1000	AL	Alabama	NaN	NaN	NaN	NaN	8008
2	1001	AL	Autauga County	2.0	2.0	2.0	2.0	62

3 rows × 34 columns

In [ ]:

```
poverty = poverty[['Stabr', "FIPS_Code", "Area_name", 'PCTPOVALL_2021']]
```

In [ ]:

```
# Merge aggregated DataFrame with the GeoDataFrame
merged_df = merged_df.merge(poverty, left_on='CountyFIPS', right_on='FIPS_Code', how='left')
merged_df.head(3)
```

Out[ ]:

	StateAbbr	CountyName	CountyFIPS	COPD_AdjPrev	ACCESS2_AdjPrev	BINGE_AdjPrev	CHECKUP_AdjPrev	CSMOKING_AdjPrev	LPA_AdjPrev	SLEEP_Ad
0	AL	Autauga	1001	6.8	10.4	15.5	76.0	16.9	29.1	
1	AL	Bullock	1011	9.8	19.2	12.4	78.2	25.7	43.6	
2	AL	Chilton	1021	8.3	14.1	15.8	72.7	21.7	34.1	

Add Education Dataframe

In [ ]:

```
educ_df = pd.read_excel("C:/Users/user/Desktop/GIS/Final Project/Education.xlsx")
educ_df.head(3)
```

Out[ ]:

	Federal Information Processing Standard (FIPS) Code	State	Area name	2003 Rural-urban Continuum Code	2003 Urban Influence Code	2013 Rural-urban Continuum Code	2013 Urban Influence Code	Less than a high school diploma, 1970	High school diploma only, 1970	Some college (1-3 years), 1970	...	Percent of adults completing some college or associate's degree, 2008-12	Percent of adults with a bachelor's degree or higher, 2008-12	Less than a high school diploma, 2017-21
0	0	US	United States	NaN	NaN	NaN	NaN	52373312.0	34158051.0	11650730.0	...	28.993579	28.484955	25050356.0
1	1000	AL	Alabama	NaN	NaN	NaN	NaN	1062306.0	468269.0	136287.0	...	29.022866	22.264896	430047.0
2	1001	AL	Autauga County	2.0	2.0	2.0	2.0	6611.0	3757.0	933.0	...	29.618142	21.707831	4126.0

3 rows × 55 columns

In [ ]:

```
educ_df = educ_df[['State', "Federal Information Processing Standard (FIPS) Code", "Percent of adults with a bachelor's degree or
```

In [ ]:

```
# Merge aggregated DataFrame with the GeoDataFrame
merged_df = merged_df.merge(educ_df, left_on='CountyFIPS', right_on='Federal Information Processing Standard (FIPS) Code', how='le
merged_df.head(3)
```

Out[ ]:

	StateAbbr	CountyName	CountyFIPS	COPD_AdjPrev	ACCESS2_AdjPrev	BINGE_AdjPrev	CHECKUP_AdjPrev	CSMOKING_AdjPrev	LPA_AdjPrev	SLEEP_Ad
0	AL	Autauga	1001	6.8	10.4	15.5	76.0	16.9	29.1	
1	AL	Bullock	1011	9.8	19.2	12.4	78.2	25.7	43.6	
2	AL	Chilton	1021	8.3	14.1	15.8	72.7	21.7	34.1	

3 rows × 21 columns

Add Population Dataframe

In [ ]:
```
pop_df = pd.read_excel("C:/Users/user/Desktop/GIS/Final Project/PopulationEstimates.xlsx")
pop_df.head(3)
```

Out[ ]:

	FIPStxt	State	Area_Name	Rural_Urban_Continuum_Code_2003	Rural_Urban_Continuum_Code_2013	Urban_Influence_2003	Urban_Influence_2013	Econon
0	0	US	United States	NaN	NaN	NaN	NaN	
1	1000	AL	Alabama	NaN	NaN	NaN	NaN	
2	1001	AL	Autauga County	2.0	2.0	2.0	2.0	

3 rows × 53 columns

In [ ]:
```
pop_df = pop_df[['State', "FIPStxt", 'R_DEATH_2021',"R_BIRTH_2021",
                 "R_INTERNATIONAL_MIG_2021", "R_DOMESTIC_MIG_2021","R_NET_MIG_2021", "R_NATURAL_CHG_2021"]]
```

In [ ]:
```
# Merge aggregated DataFrame with the GeoDataFrame
merged_df = merged_df.merge(pop_df, left_on='CountyFIPS', right_on='FIPStxt', how='left')
merged_df.head(3)
```

Out[ ]:

	StateAbbr	CountyName	CountyFIPS	COPD_AdjPrev	ACCESS2_AdjPrev	BINGE_AdjPrev	CHECKUP_AdjPrev	CSMOKING_AdjPrev	LPA_AdjPrev	SLEEP_Ad
0	AL	Autauga	1001	6.8	10.4	15.5	76.0	16.9	29.1	
1	AL	Bullock	1011	9.8	19.2	12.4	78.2	25.7	43.6	
2	AL	Chilton	1021	8.3	14.1	15.8	72.7	21.7	34.1	

3 rows × 29 columns

Add Medical Personnel Dataframe

In [ ]:
```
med_df = pd.read_csv("C:/Users/user/Desktop/GIS/Final Project/AHRF_CSV_2022-2023/DATA/ahrf2023.csv", encoding="Latin")
med_df.head(3)
```

```
C:\Users\user\AppData\Local\Temp\ipykernel_32012\3531302113.py:1: DtypeWarning: Columns (14,16,24) have mixed types. Specify dtype
option on import or set low_memory=False.
  med_df = pd.read_csv("C:/Users/user/Desktop/GIS/Final Project/AHRF_CSV_2022-2023/DATA/ahrf2023.csv", encoding="Latin")
```

Out[ ]:

	blank	fips_st_cnty	entity_file	secndry_entity_file	date_file	date_cretn	file_length	st_name	st_name_abbrev	cnty_name	...	dys_air_qulty_mesrd_21	d
0	NaN	1001	AHRF	1001	2023	23208	25907	Alabama	AL	Autauga	...	NaN	
1	NaN	1003	AHRF	1003	2023	23208	25907	Alabama	AL	Baldwin	...	280.0	
2	NaN	1005	AHRF	1005	2023	23208	25907	Alabama	AL	Barbour	...	NaN	

3 rows × 4306 columns

In [ ]:

```
med_df= med_df[["fips_st_cnty", "st_name_abbrev", "cnty_name",
                 'hosp_21', "lth_chronc_dis_21", 'stgh_tele_remote_ongong_ccm_21', "stgh_fte_phys_dent_incl_nh_21",
                 "stnglth_fte_phys_dent_incl_nh_21", "stgh_fte_rn_incl_nh_21", "stnglth_fte_rn_incl_nh_21",
                 "stgh_pharm_licd_ft_incl_nh_21", "stgh_pharm_licd_ft_incl_nh_21",
                 "stgh_resp_ther_ft_incl_nh_21", "stnglth_resp_ther_ft_incl_nh_21", "popn_est_21", "popn_mal_21",
                 "popn_est_ge65_21",
                 "popn_wh_pct_20", "popn_bl_pct_20", "popn_hsp_pct_20", "popn_asn_pct_20"]]
```

In [ ]:

```
# Merge aggregated DataFrame with the GeoDataFrame
merged_df = merged_df.merge(med_df, left_on='CountyFIPS', right_on='fips_st_cnty', how='left')
merged_df.head(3)
```

Out[ ]:

	StateAbbr	CountyName	CountyFIPS	COPD_AdjPrev	ACCESS2_AdjPrev	BINGE_AdjPrev	CHECKUP_AdjPrev	CSMOKING_AdjPrev	LPA_AdjPrev	SLEEP_Ad
0	AL	Autauga	1001	6.8	10.4	15.5	76.0	16.9	29.1	
1	AL	Bullock	1011	9.8	19.2	12.4	78.2	25.7	43.6	
2	AL	Chilton	1021	8.3	14.1	15.8	72.7	21.7	34.1	

3 rows × 50 columns

Save the Merged County Level Data to Excel File

In [ ]:

```
merged_df.to_csv("county_merged_data.csv")
```

Subset and Save the Data for West Virginia for Further Analysis

In [ ]:

```
wv_df = merged_df[merged_df["StateAbbr"]=="WV"]
wv_df.to_excel("wv_merged_data.xlsx")
```

Subset and Save the Data for New Jeresy for Further Analysis

In [ ]:

```
nj_df = merged_df[merged_df["StateAbbr"]=="NJ"]
nj_df.to_excel("nj_merged_data.xlsx")
```

In [ ]:

Lasso Regression for Feature Selection & OLS Regression

1. State Level

In [ ]:

```
merged_df_copy = merged_df
merged_df_copy.drop(columns=['resp_ther_21'], inplace=True)
merged_df_copy.dropna(inplace=True)
merged_df_copy.shape
```

Out[ ]:

```
(49, 37)
```

Lasso Regression and Cross Validation

In [ ]:

```
from sklearn.linear_model import LassoCV
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import cross_val_score, KFold
import numpy as np
import warnings
from sklearn.exceptions import ConvergenceWarning

# Define your features and target variable
features = merged_df_copy[['ACCESS2_AdjPrev', 'BINGE_AdjPrev',
       'CHECKUP_AdjPrev', 'CSMOKING_AdjPrev', 'LPA_AdjPrev', 'SLEEP_AdjPrev',
       'Unemployment_rate_2021', 'Median_Household_Income_2021',
       'PCTPOVALL_2021', "Percent of adults with a bachelor's degree or higher, 2017-21", 'phys_wkforc_21', 'rn_21', 'pharm_21', '
       'popn_pums_21', 'popn_mal_21', 'popn_50_59_21',
       'popn_ge60_21', 'popn_wh_21', 'popn_bl_21', 'popn_hsp_21',
       'popn_asn_21']]
```

```python
target = merged_df_copy['COPD_AdjPrev']

# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(features)

warnings.filterwarnings('ignore', category=ConvergenceWarning)

lasso_cv_model = LassoCV(alphas=np.logspace(-4, 4, 100), cv=10, max_iter=10000)
lasso_cv_model.fit(X_scaled, target)

# Print the optimal alpha value
print(f'Optimal alpha: {lasso_cv_model.alpha_}')

# Print the coefficients of the features
coefficients = dict(zip(features.columns, lasso_cv_model.coef_))
print("Feature coefficients:")
for feature, coefficient in coefficients.items():
    print(f'{feature}: {coefficient}')
```

```
Optimal alpha: 0.007220809018385471
Feature coefficients:
ACCESS2_AdjPrev: -0.018718161345311075
BINGE_AdjPrev: -0.3143443008132876
CHECKUP_AdjPrev: 0.07908445561143247
CSMOKING_AdjPrev: 1.0324035797734115
LPA_AdjPrev: -0.09620985608726507
SLEEP_AdjPrev: 0.2098412500591915
Unemployment_rate_2021: -0.09198218229615708
Median_Household_Income_2021: -0.14162285152197135
PCTPOVALL_2021: 0.0
Percent of adults with a bachelor's degree or higher, 2017-21: 0.12402048764632902
phys_wkforc_21: 0.0
rn_21: -0.0
pharm_21: -0.0
socwk_21: -0.08357470330998965
pt_21: -0.0
popn_pums_21: 0.0
popn_mal_21: 0.0
popn_50_59_21: 0.0
popn_ge60_21: -0.0
popn_wh_21: 0.26740978356223744
popn_bl_21: -0.23300655920494528
popn_hsp_21: 0.16526271501262682
popn_asn_21: -0.028358798207916595
```

Use the varaibles selected by Lasso, and run OLS regression again

In [ ]:

```python
import statsmodels.api as sm

def run_regression(y, X):
    # Add a constant term to the independent variables
    X = sm.add_constant(X)

    # Fit the linear regression model
    model = sm.OLS(y, X).fit()

    # Print results
    print("Target variable:", y.name)
    print(model.summary())
    print("\n")

y = merged_df_copy['COPD_AdjPrev']
X = merged_df_copy[['ACCESS2_AdjPrev', 'BINGE_AdjPrev',
       'CHECKUP_AdjPrev', 'CSMOKING_AdjPrev', 'LPA_AdjPrev', 'SLEEP_AdjPrev',
       'Unemployment_rate_2021', 'Median_Household_Income_2021',
       "Percent of adults with a bachelor's degree or higher, 2017-21",'popn_wh_21', 'popn_bl_21', 'popn_hsp_21',
       'popn_asn_21']]
run_regression(y, X)
```

```
Target variable: COPD_AdjPrev
                            OLS Regression Results                            
==============================================================================
Dep. Variable:           COPD_AdjPrev   R-squared:                       0.945
Model:                            OLS   Adj. R-squared:                  0.925
Method:                 Least Squares   F-statistic:                     46.45
Date:                Wed, 15 May 2024   Prob (F-statistic):           3.77e-18
Time:                        20:35:57   Log-Likelihood:                -11.010
No. Observations:                  49   AIC:                             50.02
Df Residuals:                      35   BIC:                             76.51
Df Model:                          13                                         
Covariance Type:            nonrobust                                         
=================================================================================================================================
                                                                   coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------------------------------------------------
const                                                            1.3264      2.032      0.653      0.518      -2.799       5.452
ACCESS2_AdjPrev                                                 -0.0202      0.032     -0.633      0.531      -0.085       0.045
BINGE_AdjPrev                                                   -0.1537      0.032     -4.865      0.000      -0.218      -0.090
CHECKUP_AdjPrev                                                  0.0203      0.021      0.961      0.343      -0.023       0.063
CSMOKING_AdjPrev                                                 0.3335      0.046      7.209      0.000       0.240       0.427
LPA_AdjPrev                                                     -0.0454      0.033     -1.356      0.184      -0.113       0.023
SLEEP_AdjPrev                                                    0.0919      0.034      2.721      0.010       0.023       0.161
Unemployment_rate_2021                                          -0.1175      0.061     -1.921      0.063      -0.242       0.007
Median_Household_Income_2021                                  -1.86e-05   1.26e-05     -1.478      0.148   -4.42e-05    6.96e-06
Percent of adults with a bachelor's degree or higher, 2017-21    0.0229      0.017      1.369      0.180      -0.011       0.057
popn_wh_21                                                    8.057e-08   3.94e-08      2.043      0.049       5e-10    1.61e-07
popn_bl_21                                                   -3.229e-07   1.39e-07     -2.320      0.026   -6.06e-07   -4.04e-08
popn_hsp_21                                                   1.449e-07   7.17e-08      2.021      0.051    -6.5e-10     2.9e-07
popn_asn_21                                                  -2.783e-07      2e-07     -1.388      0.174   -6.85e-07    1.29e-07
==============================================================================
Omnibus:                        2.864   Durbin-Watson:                   2.286
Prob(Omnibus):                  0.239   Jarque-Bera (JB):                2.415
Skew:                          -0.543   Prob(JB):                        0.299
Kurtosis:                       2.949   Cond. No.                     2.21e+08
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.21e+08. This might indicate that there are
strong multicollinearity or other numerical problems.
```

2. West Virginia

In [ ]:

```python
# Define your features and target variable
features = wv_df[['ACCESS2_AdjPrev', 'BINGE_AdjPrev', 'CHECKUP_AdjPrev',
                      'CSMOKING_AdjPrev', 'LPA_AdjPrev', 'SLEEP_AdjPrev',
                      'Unemployment_rate_2021', 'Median_Household_Income_2021',
                      'PCTPOVALL_2021', "Percent of adults with a bachelor's degree or higher, 2017-21",
                      'R_INTERNATIONAL_MIG_2021', 'R_DOMESTIC_MIG_2021',
                      'R_NET_MIG_2021', 'hosp_21', 'lth_chronc_dis_21', 'stgh_tele_remote_ongong_ccm_21',
                      'stgh_fte_phys_dent_incl_nh_21', 'stnglth_fte_phys_dent_incl_nh_21', 'stgh_fte_rn_incl_nh_21',
                      'stnglth_fte_rn_incl_nh_21', 'stgh_pharm_licd_ft_incl_nh_21', 'stgh_pharm_licd_ft_incl_nh_21',
                      'stgh_resp_ther_ft_incl_nh_21', 'stnglth_resp_ther_ft_incl_nh_21', 'popn_est_21', 'popn_mal_21',
                      'popn_est_ge65_21', 'popn_wh_pct_20', 'popn_bl_pct_20', 'popn_hsp_pct_20', 'popn_asn_pct_20']]

target = wv_df['COPD_AdjPrev']
```

In [ ]:

```python
from sklearn.linear_model import LassoCV
from sklearn.preprocessing import StandardScaler

# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(features)

# Fit Lasso regression model
warnings.filterwarnings('ignore', category=ConvergenceWarning)

lasso_cv_model = LassoCV(alphas=np.logspace(-4, 4, 100), cv=10, max_iter=10000)
lasso_cv_model.fit(X_scaled, target)

# Print the optimal alpha value
print(f'Optimal alpha: {lasso_cv_model.alpha_}')

# Print the coefficients of the features
coefficients = dict(zip(features.columns, lasso_cv_model.coef_))
print("Feature coefficients:")
for feature, coefficient in coefficients.items():
    print(f'{feature}: {coefficient}')
```

```
Optimal alpha: 0.010476157527896652
Feature coefficients:
ACCESS2_AdjPrev: 0.32189635027036073
BINGE_AdjPrev: -0.27037167493493136
CHECKUP_AdjPrev: -0.028072920467482238
CSMOKING_AdjPrev: 0.5316574031772938
LPA_AdjPrev: 0.31210994438258477
SLEEP_AdjPrev: 0.01809328590734056
Unemployment_rate_2021: 0.03707485263194225
Median_Household_Income_2021: -0.0
PCTPOVALL_2021: 0.0
Percent of adults with a bachelor's degree or higher, 2017-21: 0.0
R_INTERNATIONAL_MIG_2021: -0.03149060679583304
R_DOMESTIC_MIG_2021: -0.0
R_NET_MIG_2021: -0.010408793653726812
hosp_21: -0.0
lth_chronc_dis_21: 0.0
stgh_tele_remote_ongong_ccm_21: -0.0
stgh_fte_phys_dent_incl_nh_21: -0.12404213275998197
stnglth_fte_phys_dent_incl_nh_21: 0.0
stgh_fte_rn_incl_nh_21: -0.0
stnglth_fte_rn_incl_nh_21: 0.06907272537189774
stgh_pharm_licd_ft_incl_nh_21: -0.0
stgh_resp_ther_ft_incl_nh_21: 0.0
stnglth_resp_ther_ft_incl_nh_21: 0.0
popn_est_21: 0.10751157540234069
popn_mal_21: 0.0
popn_est_ge65_21: 0.0
popn_wh_pct_20: 0.0
popn_bl_pct_20: -0.13354044693680733
popn_hsp_pct_20: -0.00518444534100251
popn_asn_pct_20: -0.0
```

In [ ]:

```python
import statsmodels.api as sm

def run_regression(y, X):
    # Add a constant term to the independent variables
    X = sm.add_constant(X)

    # Fit the linear regression model
    model = sm.OLS(y, X).fit()

    # Print results
    print("Target variable:", y.name)
    print(model.summary())
    print("\n")

y = wv_df['COPD_AdjPrev']
X = wv_df[['CSMOKING_AdjPrev', "ACCESS2_AdjPrev", 'LPA_AdjPrev', "BINGE_AdjPrev", "CHECKUP_AdjPrev", "SLEEP_AdjPrev",
    "stgh_fte_phys_dent_incl_nh_21", "Unemployment_rate_2021", "R_INTERNATIONAL_MIG_2021", "R_NET_MIG_2021",
    'stnglth_fte_rn_incl_nh_21', 'popn_est_21', 'stgh_resp_ther_ft_incl_nh_21',
    'popn_bl_pct_20', 'popn_hsp_pct_20']]
run_regression(y, X)
```

```
Target variable: COPD_AdjPrev
                            OLS Regression Results
==============================================================================
Dep. Variable:           COPD_AdjPrev   R-squared:                       0.989
Model:                            OLS   Adj. R-squared:                  0.984
Method:                 Least Squares   F-statistic:                     226.0
Date:                Wed, 15 May 2024   Prob (F-statistic):           4.58e-33
Time:                        20:39:01   Log-Likelihood:                 25.961
No. Observations:                  55   AIC:                            -19.92
Df Residuals:                      39   BIC:                             12.20
Df Model:                          15
Covariance Type:            nonrobust
=================================================================================================
                                  coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------------------------
const                           9.1614      2.897      3.163      0.003       3.303      15.020
CSMOKING_AdjPrev                0.1716      0.043      4.016      0.000       0.085       0.258
ACCESS2_AdjPrev                 0.3095      0.088      3.508      0.001       0.131       0.488
LPA_AdjPrev                     0.0741      0.032      2.281      0.028       0.008       0.140
BINGE_AdjPrev                  -0.3943      0.068     -5.785      0.000      -0.532      -0.256
CHECKUP_AdjPrev                -0.0428      0.027     -1.586      0.121      -0.097       0.012
SLEEP_AdjPrev                   0.0070      0.018      0.384      0.703      -0.030       0.044
stgh_fte_phys_dent_incl_nh_21  -0.0034      0.001     -3.901      0.000      -0.005      -0.002
Unemployment_rate_2021          0.0250      0.024      1.038      0.306      -0.024       0.074
R_INTERNATIONAL_MIG_2021       -0.1321      0.120     -1.104      0.277      -0.374       0.110
R_NET_MIG_2021                 -0.0031      0.005     -0.624      0.536      -0.013       0.007
stnglth_fte_rn_incl_nh_21       0.0023      0.001      2.244      0.031       0.000       0.004
popn_est_21                  8.618e-06    2.3e-06      3.740      0.001    3.96e-06    1.33e-05
stgh_resp_ther_ft_incl_nh_21   -0.0021      0.003     -0.720      0.476      -0.008       0.004
popn_bl_pct_20                 -0.0605      0.014     -4.309      0.000      -0.089      -0.032
popn_hsp_pct_20                -0.0274      0.041     -0.660      0.513      -0.111       0.057
==============================================================================
Omnibus:                        1.029   Durbin-Watson:                   2.245
Prob(Omnibus):                  0.598   Jarque-Bera (JB):                1.043
Skew:                          -0.205   Prob(JB):                        0.594
Kurtosis:                       2.464   Cond. No.                     5.54e+06
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 5.54e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [ ]:

3. New Jeresy

In [ ]:

```
# Define your features and target variable
features = nj_df[['ACCESS2_AdjPrev', 'BINGE_AdjPrev', 'CHECKUP_AdjPrev',
                      'CSMOKING_AdjPrev', 'LPA_AdjPrev', 'SLEEP_AdjPrev',
                      'Unemployment_rate_2021', 'Median_Household_Income_2021',
                      'PCTPOVALL_2021', "Percent of adults with a bachelor's degree or higher, 2017-21",
                      'R_INTERNATIONAL_MIG_2021', 'R_DOMESTIC_MIG_2021',
                      'R_NET_MIG_2021', 'hosp_21', 'lth_chronc_dis_21', 'stgh_tele_remote_ongong_ccm_21',
                      'stgh_fte_phys_dent_incl_nh_21', 'stnglth_fte_phys_dent_incl_nh_21', 'stgh_fte_rn_incl_nh_21',
                      'stnglth_fte_rn_incl_nh_21', 'stgh_pharm_licd_ft_incl_nh_21', 'stgh_pharm_licd_ft_incl_nh_21',
                      'stgh_resp_ther_ft_incl_nh_21', 'stnglth_resp_ther_ft_incl_nh_21', 'popn_est_21', 'popn_mal_21',
                      'popn_est_ge65_21', 'popn_wh_pct_20', 'popn_bl_pct_20', 'popn_hsp_pct_20', 'popn_asn_pct_20']]

target = nj_df['COPD_AdjPrev']

# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(features)

warnings.filterwarnings('ignore', category=ConvergenceWarning)

lasso_cv_model = LassoCV(alphas=np.logspace(-4, 4, 100), cv=20, max_iter=10000)
lasso_cv_model.fit(X_scaled, target)

# Print the optimal alpha value
print(f'Optimal alpha: {lasso_cv_model.alpha_}')

# Print the coefficients of the features
coefficients = dict(zip(features.columns, lasso_cv_model.coef_))
print("Feature coefficients:")
for feature, coefficient in coefficients.items():
    print(f'{feature}: {coefficient}')
```

```
Optimal alpha: 0.04641588833612782
Feature coefficients:
ACCESS2_AdjPrev: 0.0
BINGE_AdjPrev: -0.0
CHECKUP_AdjPrev: -0.0
CSMOKING_AdjPrev: 0.4744816501800917
LPA_AdjPrev: 0.07542891675057854
SLEEP_AdjPrev: 0.0
Unemployment_rate_2021: 0.0
Median_Household_Income_2021: -0.0
PCTPOVALL_2021: 0.0
Percent of adults with a bachelor's degree or higher, 2017-21: -0.2223564934504932
R_INTERNATIONAL_MIG_2021: -0.0
R_DOMESTIC_MIG_2021: 0.0
R_NET_MIG_2021: 0.0
hosp_21: -0.0
lth_chronc_dis_21: 0.0
stgh_tele_remote_ongong_ccm_21: -0.0
stgh_fte_phys_dent_incl_nh_21: -0.0
stnglth_fte_phys_dent_incl_nh_21: -0.0
stgh_fte_rn_incl_nh_21: 0.0
stnglth_fte_rn_incl_nh_21: 0.0
stgh_pharm_licd_ft_incl_nh_21: 0.0
stgh_resp_ther_ft_incl_nh_21: 0.0
stnglth_resp_ther_ft_incl_nh_21: 0.0
popn_est_21: -0.0
popn_mal_21: -0.0
popn_est_ge65_21: -0.0
popn_wh_pct_20: 0.0
popn_bl_pct_20: 0.0
popn_hsp_pct_20: -0.0
popn_asn_pct_20: -0.08309377488977375
```

In [ ]:

```python
import statsmodels.api as sm

def run_regression(y, X):
    # Add a constant term to the independent variables
    X = sm.add_constant(X)

    # Fit the linear regression model
    model = sm.OLS(y, X).fit()

    # Print results
    print("Target variable:", y.name)
    print(model.summary())
    print("\n")

y = nj_df['COPD_AdjPrev']
X = nj_df[['CSMOKING_AdjPrev', "LPA_AdjPrev", "Percent of adults with a bachelor's degree or higher, 2017-21", "popn_asn_pct_20"]]
run_regression(y, X)
```

```
Target variable: COPD_AdjPrev
                            OLS Regression Results                            
==============================================================================
Dep. Variable:           COPD_AdjPrev   R-squared:                       0.973
Model:                            OLS   Adj. R-squared:                  0.967
Method:                 Least Squares   F-statistic:                     146.1
Date:                Wed, 15 May 2024   Prob (F-statistic):           2.24e-12
Time:                        20:41:02   Log-Likelihood:                 11.456
No. Observations:                  21   AIC:                            -12.91
Df Residuals:                      16   BIC:                            -7.690
Df Model:                           4                                         
Covariance Type:            nonrobust                                         
==================================================================================================================================
                                                                    coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------------------------------------------------------
const                                                             2.2098      1.101      2.007      0.062      -0.124       4.544
CSMOKING_AdjPrev                                                  0.2140      0.043      5.000      0.000       0.123       0.305
LPA_AdjPrev                                                       0.0338      0.016      2.076      0.054      -0.001       0.068
Percent of adults with a bachelor's degree or higher, 2017-21    -0.0165      0.012     -1.409      0.178      -0.041       0.008
popn_asn_pct_20                                                  -0.0207      0.008     -2.618      0.019      -0.037      -0.004
==============================================================================
Omnibus:                        0.067   Durbin-Watson:                   2.607
Prob(Omnibus):                  0.967   Jarque-Bera (JB):                0.204
Skew:                           0.113   Prob(JB):                        0.903
Kurtosis:                       2.574   Cond. No.                     1.54e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.54e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Plotting used in Slides

In [ ]:

```python
import seaborn as sns
import matplotlib.pyplot as plt


# Sort the DataFrame by COPD_AdjPrev in descending order and select the first ten rows
state_order = merged_df.sort_values(by='COPD_AdjPrev', ascending=False)

# Plotting
plt.figure(figsize=(12, 6))
sns.barplot(x='StateAbbr', y='COPD_AdjPrev', data=state_order, palette='rocket')
plt.xlabel('State')
plt.ylabel('COPD Adjusted Prevalence Rate')
plt.title('COPD Adjusted Prevalence Rate by State')
plt.xticks(rotation=45, ha='right')

# Adding annotations using bar_label
ax = sns.barplot(data=state_order, x='StateAbbr', y='COPD_AdjPrev', palette='rocket')
for p in ax.patches:
    ax.annotate(f"{p.get_height():.1f}", (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=8, color='black', xytext=(0, 5),
                textcoords='offset points')


plt.tight_layout()
plt.show()
```

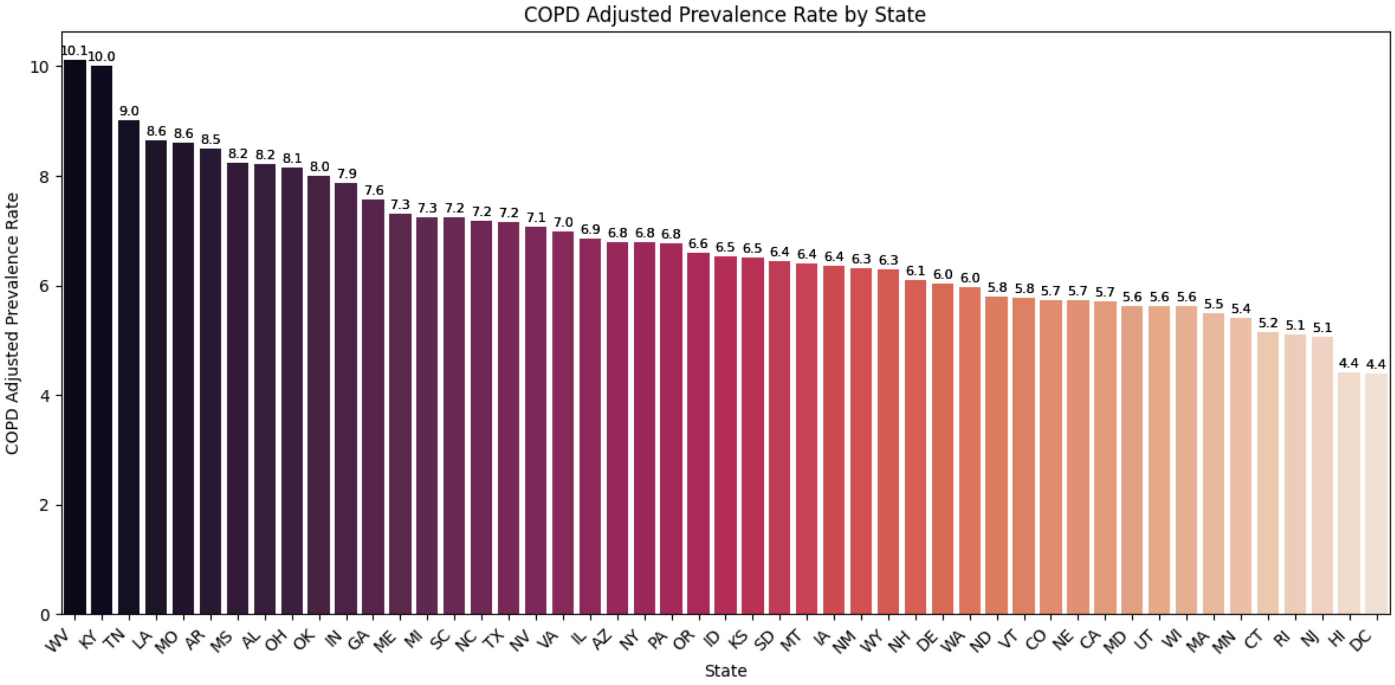
```
C:\Users\user\AppData\Local\Temp\ipykernel_23228\2082703754.py:10: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `
legend=False` for the same effect.

  sns.barplot(x='StateAbbr', y='COPD_AdjPrev', data=state_order, palette='rocket')
C:\Users\user\AppData\Local\Temp\ipykernel_23228\2082703754.py:17: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `
legend=False` for the same effect.

  ax = sns.barplot(data=state_order, x='StateAbbr', y='COPD_AdjPrev', palette='rocket')
```



In [ ]:

```python
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming wv_df is your DataFrame containing COPD data
# Sort the DataFrame by COPD_AdjPrev in descending order and select the first ten rows
top_10_counties = wv_df.sort_values(by='COPD_AdjPrev', ascending=False).head(10)

# Plotting
plt.figure(figsize=(10, 6))
sns.barplot(x='CountyName', y='COPD_AdjPrev', data=top_10_counties, palette='viridis')
plt.xlabel('County')
plt.ylabel('COPD Adjusted Prevalence Rate')
plt.title('Top 10 Counties in West Virginia by COPD Adjusted Prevalence Rate')
plt.xticks(rotation=45, ha='right')

# Adding annotations using bar_label
ax = sns.barplot(data=top_10_counties, x='CountyName', y='COPD_AdjPrev', palette='rocket')
for p in ax.patches:
    ax.annotate(f"{p.get_height():.2f}", (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=8, color='black', xytext=(0, 5),
                 textcoords='offset points')


plt.tight_layout()
plt.show()
```

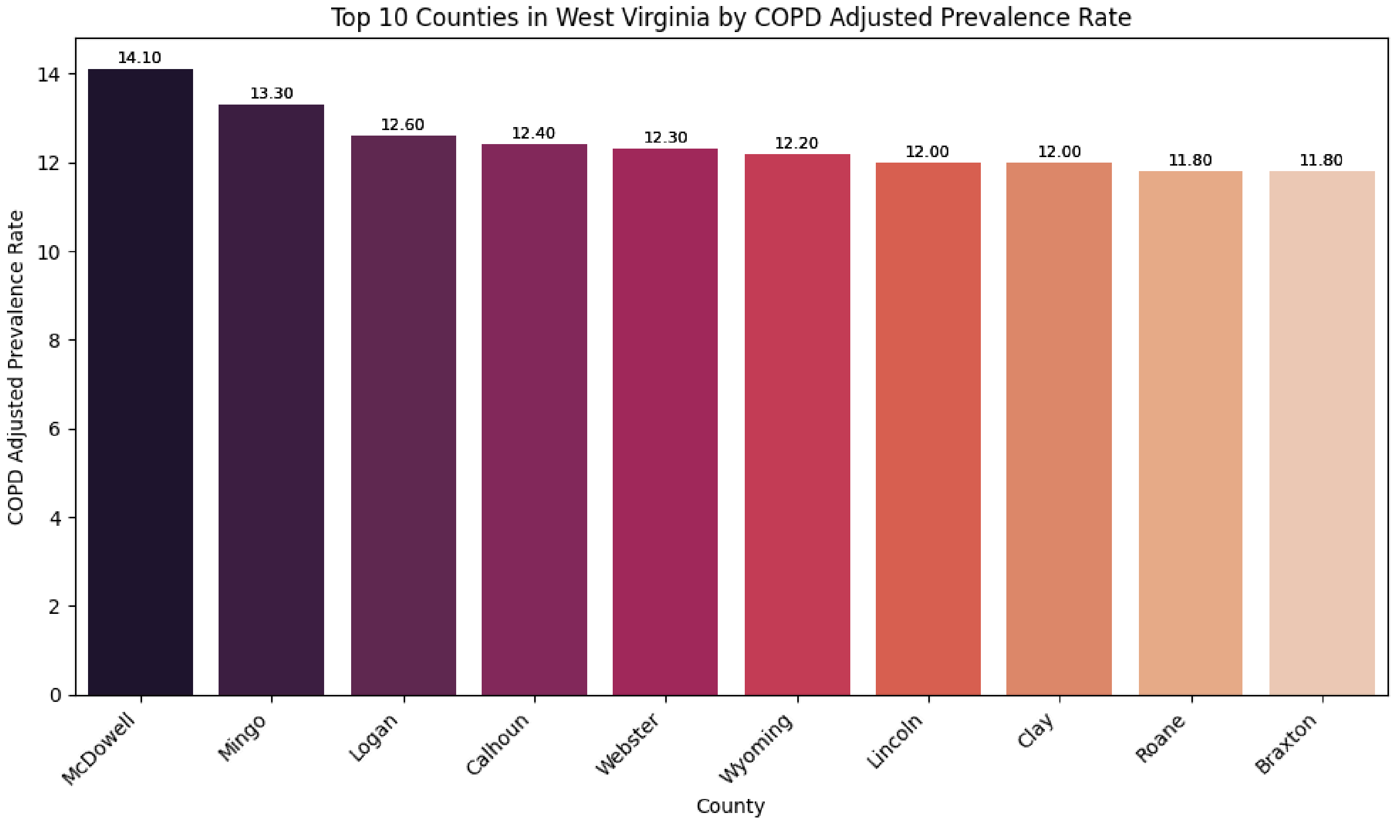
```
C:\Users\user\AppData\Local\Temp\ipykernel_23228\3043263568.py:10: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `
legend=False` for the same effect.

  sns.barplot(x='CountyName', y='COPD_AdjPrev', data=top_10_counties, palette='viridis')
C:\Users\user\AppData\Local\Temp\ipykernel_23228\3043263568.py:17: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `
legend=False` for the same effect.

  ax = sns.barplot(data=top_10_counties, x='CountyName', y='COPD_AdjPrev', palette='rocket')
```


Top 10 Counties in West Virginia by COPD Adjusted Prevalence Rate
COPD Adjusted Prevalence Rate
McDowell
Mingo
Logan
Calhoun
Webster
Wyoming
Lincoln
Clay
Roane
Braxton
County