```
# Python for Healthcare

## Hospital Spending Narrative

### Import Standard Libraries
import os # Inlcuded in every script DC!

### Set working directory
os.chdir("C:/Users/drewc/GitHub/python-for-healthcare/pymodule1") # Set wd to
project repository

#################### Break ####################

# Section A: 2018 MSPB by State (EDA)

## Step 1: Import Libraries and Data

### Import Libraries
import pandas as pd # Incldued in every code script for DC!
import numpy as np # Incldued in every code script for DC!

import geopandas as gp # Simple mapping with pandas like syntax
import matplotlib.pyplot as plt # Comprehensive graphing package in python

### Import CMS Data
df_cms = pd.read_csv("_data/health_mspb_hospital_stage.csv", encoding =
"ISO-8859-1") # Import dataset saved as csv in _data folder

### Import State Shape File
gdf_state = gp.read_file("_data/health_maps_state_stage.shp")

### Verify CMS
df_cms.info() # Get class, memory, and column info: names, data types, obs.
df_cms.head() # Print first 5 observations

## Step 2: Prepare Data for Analysis

### Select only State and Measure
df_filter = df_cms.filter(["State", "Score"]) # Keep only selected columns

### Group by State
df_group = df_filter.groupby(["State"], as_index = False).mean() # Group data By
Columns and Sum

### Rename Score as MSPB
df_rename = df_group.rename(columns = {"Score": "MSPB"}) # Rename column

### Drop NA values
df_na = df_rename.dropna() # Drop all rows with NA values
```

```
### Rename Dataframe
df_mspb = df_na # Rename sorted dataframe as MSPB for clarity

### Verify MSPB
df_mspb.info() # Get class, memory, and column info: names, data types, obs.
df_mspb.head() # Print first 5 observations

## Step 3: Conduct Analysis and Tests

### Summary Statistics for States
df_mspb.describe() # Get summary statistics for numerical columns in data frame

### Top 5 States for MSPB
df_mspb = df_mspb.sort_values(by = ["MSPB"], ascending = False) # Sort Columns by
Value

### Verify MSPB
df_mspb.head() # Print first 5 observations

## Step 4: Create Visuals and Outputs

### Inner Join State and Geometry
gdf_join = pd.merge(gdf_state, df_mspb, on = "State", how = "inner") # Join by
column while keeping only items that exist in both, select outer or left for other
options

### Create Map fig
gdf_join.plot(column = "MSPB", cmap = "Blues", legend = False).set_axis_off() #
Create simple choropleth map in geopandas

## Verify
plt.show() # Show created plots

# End Section
print("THE END") # Print result

##################### Break ####################

# MSPB by State and Money (Q-Q)
print("Section Start") # Print result

## Step 1: Import Libraries and Data

### Import Statistics Packages
import statsmodels.api as sm # Regression modeling in scipy

### Import Money Data
df_money = pd.read_csv("_data/health_money_state_stage.csv", encoding =
"ISO-8859-1") # Import dataset saved as csv in _data folder
```

```
### Verify Money
df_money.info() # Get class, memory, and column info: names, data types, obs.
df_money.head() # Print first 5 observations

## Step 2: Prepare Data for Analysis

### Inner MSPB
df_join = pd.merge(df_mspb, df_money, on = "State", how = "inner") # Join by column
while keeping only items that exist in both, select outer or left for other options
df_join.info() # Get class, memory, and column info: names, data types, obs.

### Drop Values with NA
df_na = df_join.dropna() # Drop all rows with NA values, 0 = rows, 1 = columns

### Drop State
df_drop = df_na.drop(columns = ["State"]) # Drop Unwanted Columns

### Rename to Regression
df_reg = df_drop

### Verify MSPB
df_reg.info() # Get class, memory, and column info: names, data types, obs.
df_reg.head() # Print first 5 observations

## Step 3: Conduct Analysis and Tests

### Linear Regression Model
features = df_reg.columns.drop(["MSPB"]) # Drop outcome variable and Geo to isolate
all predictor variable names as features
x = df_reg[features] # features as x
y = df_reg["MSPB"] # Save outcome variable as y
model = sm.OLS(y, x).fit() # Run Linear Regression Model This may but most likely
wont take time
result = model.summary() # Create Summary of final model

### Verify Regression
print(result) # Print result to verify

## Step 4: Create Visuals and Outputs

### Inner Join State and Geometry
gdf_join = pd.merge(gdf_state, df_na, on = "State", how = "inner") # Join by column
while keeping only items that exist in both, select outer or left for other options

### Create Map fig
fig, (ax1, ax2, ax3) = plt.subplots(ncols = 3)

### Create 1st axis
gdf_join.plot(column = "MDIncome", cmap = "Greens", ax = ax1, legend =
False).set_axis_off()
```

```
ax1.set_title("Mean Physician Income")

### Create 2nd axis
gdf_join.plot(column = "ForProfit", cmap = "Reds", ax = ax2, legend =
False).set_axis_off()
ax2.set_title("Percent of Hospitals that are For Profit")

### Create 3rd axis
gdf_join.plot(column = "MSPB", cmap = "Blues", ax = ax3, legend =
False).set_axis_off() # Create simple choropleth map in geopandas
ax3.set_title("Mean MSPB")

### Save to figure file
fig.suptitle("For Profit Hospitals, Physician Income, and MSPB in 2018")
fig.savefig("_fig/health_money_state_map.jpeg", bbox_inches = "tight")

## Verify
plt.show() # Show created plots

# End Section
print("THE END") # Print result
```