Notes

- The point breakdowns below assume each assignment is out of 100. Although the assignments are worth more "points" on Canvas, it's easier to think in terms of a percentage scale from 0%-100%.
- Since students had the option to complete either 2 or 3 RQs, the notes about points account for this. Essentially, I allocate a certain number of points to the analysis parts of the assignments. I split those into 2 if the student had 2 RQs, and split those points into 3 if the student had 3 RQs.

# 1 Final Report

1. Summary (10 points)
2. Problem diagnosis (10 points)
3. Research questions (6 points)
4. Analysis for RQ1 (27 points if 2 RQs, 18 points if 3 RQs)
5. Analysis for RQ2 (27 points if 2 RQs, 18 points if 3 RQs)
6. Analysis for RQ3 (27 points if 2 RQs, 18 points if 3 RQs)
7. Conclusion: Recommendations (20 points)
    a. Since the assignment asks for 2 recommendations, each recommendation is worth 10 points.

## 2 Analysis

For the analysis, each RQ is graded independently.

Correctness (30 points per RQ if 2 RQs, 20 points per RQ if 3 RQs)

- Your work must be completed inside the Vocareum notebook in python. All results must be contained to the notebook (e.g., no links to visualizations on external websites).
- You are allowed to use any data sources, python libraries, and analysis methods.
- Your code should be free of bugs. It should be possible for the grader to run your scripts and recreate your results.
- You should use appropriate libraries/functions based on your RQs. You should implement these functions correctly.
- Your code should correctly implement the steps you describe in your comments. The outputs of your code should match the outputs that you describe generating.
- Your analysis should consider the robustness of your results.
    - If your analysis involves aggregating data, you should consider whether different levels of aggregation affect your results.
    - If you made judgment calls about how to define metrics or outcomes, you should consider whether different definitions affect your results.
    - In general, you should be thinking about: 1) Are there alternative explanations for my results? 2) Can I test whether the data supports or contradicts these alternative explanations?

Clarity of code (20 points per RQ if 2 RQs, 13.5 points per RQ if 3 RQs)

- Your code should be well-organized and should include comments explaining what each code block is for. In other words, your code should be readable by someone else (in particular: a GSI should be able to review your submission and easily understand what you're doing). Here are some guidelines for code organization:
    - Jupyter notebook cells should be broken into digestible chunks.
    - Variable and function names should clearly indicate what they refer to.
    - Indentation should clarify the hierarchy and flow of your code.
    - Your code should follow consistent patterns for labeling, indentation, etc.
    - Take advantage of built-in python and pandas functionality where appropriate.
- It is particularly important that you explain your reasoning and justification for the steps you take. This is particularly important when you are making judgment calls such as dropping observations or defining metrics.
- Keep in mind that clarity of code can also support your score for correctness. The better you explain what you're doing, the more likely that the grader will understand why your analysis is appropriate.

## 3 Reflection

The reflection will be graded holistically. The points are not broken down into discrete categories. Your work should demonstrate that you have spent real time thinking about lessons learned. Your reflection should describe what you learned, how you learned that lesson, and how that lesson will inform your work in the future.