Capstone Project Summary

Daniel J Matthews

- **Data Cleaning**
  - Downloaded the dataset from data.gov
  - Preliminary cleaning:
    - Strips whitespace from beginning and end of column names, replaces spaces in column names with underscores and makes all characters lower case. Defining columns this way makes writing code easier in the notebook.
    - All dollar amounts were given with dollar signs so pandas read them as strings. For these columns it strips the dollar sign makes them float values so they can be manipulated mathematically.
    - Zip codes with leading zeros were in the csv files as 4 digit numbers, so this pads the zip code to a 5 character string instead of an integer. Zip codes are never used mathematically so it's better to have them as strings.
  - Since the drg_definition column is a very long string it would be cumbersome to write out the long string every time you wanted to filter by procedure. To avoid this, the following code inserts a string type column at the beginning of the dataframe with just the three digit code describing the procedure (which is essentially the first three string characters of the drg_definition).
  - All city names that were longer than 15 characters were cut off at 15. This needed to be corrected to improve the geocoding results.
    - The steps taken to correct the city names:
      1. Select only the rows with unique city names, filter on all cities with 15 characters and write them to a csv file.
      2. Edit the cvs file by hand.
         - The previous step generated a list of 70 city names. After removing the cities whose names were exactly 15 characters and therefore not truncated, 46 city names remained which is feasible to just edit by hand.
         - For each of the 46 truncated names, a second column is added that contains the corresponding full city name. Column 1 is labeled 'city' and column 2 is labeled 'city_corrected', and the file name is city_corrections.csv.
      3. For each row of the city_corrections.csv file we iterate through the entire dataset and replace every occurance of the truncated city name with the full city name.

- **Geocoding**
  - In order to perform any analysis using the provider location in the United States it is necessary to find the GPS locations of each provider. Using the GoogleMaps API I found the GPS coordinates for every provider in the IPPS dataset.
  - In order to query the GoogleMaps API the address must be given as a single string containing the full address. The following steps create a dataframe that is suitable for querying the API.
    - I combined the address information into a single string (‘provider_name’, ’provider_street_address’, ‘provider_city’, ‘provider_state’, ‘provider_zip_code’).
    - I found that some characters create problems when querying the GoogleMaps API so I remove all commas and apostrophes from the address strings. Some of the provider names also ended with ", THE", so I remove those as well.

        - Sometimes the GPS coordinates returned by GoogleMaps was more accurate when including ‘provider_name’, but most of the time not using the provider name gave the best results.
            - Because of this I ran the geocoding for both cases. I plotted the difference in latitude and longitude between the two results, and for each case that had a very large difference I checked it on the google maps website to see which on was correct.
            - I found that for the cases where the difference was very large, sometimes including the provider name gave the best results and sometimes not using the name was best. As the difference got smaller, NOT including the provider name consistently gave the better results.
        - The best results for each provider were combined and put into a single csv file
        - I created a heatmap of the provider locations overlaid on a GoogleMaps map.

- **Exploratory Data Analysis (EDA)**
    - This is to get a general idea of the kinds of numbers in the dataset, such as the number of rows, the number of providers that report data for a given DRG, etc.
    - Numbers
        - Number of rows: 163,065
        - Number of providers: 3,337
        - Number of different procedure ID’s: 100 (not every provider had all 100)
        - Number of discharges: 6,975,318
    - Calculations
        - The number of providers that reported a given number of DRG definitions. For example, 95 providers only reported data for a single DRG definition, and 48 providers reported data for all 100 DRGs. (bar plot)
        - The total number of providers nationwide that reported data for each DRG definition (bar plot)
        - Plot of average covered charges and average total payments for procedure 039 for all providers. Total payments has small scatter, total cost has very large scatter (scatter plot)
        - National median charges, median payments and median Medicare payments for each procedure type (bar plot)
        - For each row calculated the fractional difference in cost, payments and Medicare payments compared to the national medians for that procedure (histogram)
        - Looked at whether a provider had more procedures above the national median or more procedures below for average total billed charges, and plotted its location (GPS coordinates) (scatter plot)
            - Made separate plots for each case (more above national median or more below)
            - It appears that providers with more procedures above national median are more tightly clustered near large cites.
            - Providers with more procedures below the national median are more evenly distributed.
        - Averaged the fractional differences for each provider over all procedures (scatter plot)
            - Saw some variation by state

- **Machine Learning: Random Forest Classification**
    - I use a random forest classifier to predict whether or not the average billed charge for a given DRG definition at a given provider will be above or below the national median. I split

the data into the test set and training set and use the training set to train the model and then test the model by making predictions on the test set. The predictions are compared to the actual data to see how well the model performs.

- I added a column classifying the procedure at that provider as being above or below the national median.
- Using the GPS coordinates as the input X and the fractional difference classifier as y with the Random Forest Classifier, I performed a test-train split with the training set as 2/3 of the total data set.
- I obtained a R^2 score of 0.875 when using the training model on the test set
- I plotted the confusion matrix for the results, and it was a symmetrical matrix with ~0.87-0.88 on the diagonal and ~0.12-0.13 on the off-diagonals.
- I calculated an AUC score of about 0.95
- **Adding in other variables such as the population and median income by zip code seemed to have no significant effect. I also calculated the number of other providers within 40 miles of each provider, but adding that variable had no effect as well.**
- **I also found that I get the same results with just the provider_id tag as the input X using pandas' get_dummies to convert the variable to an indicator variable rather than numerical value. This makes me think that the GPS coordinates are essentially acting as an indicator for the specific provider rather than giving any useful information as far as relative position to other hospitals.**