ACasaccio - Assignment 10.3 - Final Project Step 2

Alysen Casaccio

2023-11-04

```
#install.packages("readxl")
#install.packages("dplyr")
#install.packages("stringr")
#install.packages("tidyr")
#install.packages("ggplot2")
library(readxl)
```

```
## Warning: package 'readxl' was built under R version 4.3.2
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.3.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(stringr)
```

```
## Warning: package 'stringr' was built under R version 4.3.2
```

```
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 4.3.2
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

Introduction:

In the healthcare industry today, understanding clinician quality of care key performance metrics and the meaning behind these measurements, which are also being submitted at a nationwide scale, is essential for any Ambulatory Quality Director to draw conclusions about their clinics and clinicians. These conclusions help them select the best possible interventions to drive results better patient outcomes across their particular clinicians, clinics, and health systems.

Comparing internal results to those of external, national feedback can also be helpful, but this becomes a data science problem due to the large amount of publicly reported data available, which can often be overwhelming to new Quality Directors. I will be comparing a dataset of de-identified, internal health system results to broader, national results, leveraging identical key performance indicators (KPI) wherever possible to better gauge the internal practices' performance, identify areas of improvement, and develop target goals for each KPI that are specific, achievable, and relevant for the upcoming performance year.

Research Questions:

1. How does the performance of our clinicians compare to national benchmarks on specific KPI?
2. Are there patterns of higher or lower performance associated with specific locations, measures, or clinician roles?
3. What is the relationship between the role of a clinician (MD, NP, PA) and their performance on various KPI?
4. Is there any correlation between practice locations and clinician performance?
5. How does the performance vary across different KPI and what might be driving these variations?
6. Are there any outlier clinicians or practices that are significantly under- or over-performing?
7. What are the common characteristics of top-performing clinicians or practices?
8. How can the insights gained from this analysis be translated into actionable strategies for quality improvement, including specific, achievable, and relevant target goals for the upcoming year to improve clinician performance?

Approach:

How to Import and Clean My Data

Data Import I imported two excel worksheets into RStudio and performed some initial cleaning and transformation. The first worksheet is 3 quarters of performance against quality of care measurements for small primary care practices (less than 15 clinicians each). The second worksheet has the CMS established national benchmarks and decile boundaries for the measurements being performed. Prior to loading and reading the data, I de-identified clinician names and practice locations. Although there were no HIPAA concerns, this internal dataset does contain sensitive clinician information, so while I could have done this as a part of my data cleaning, using 'mutate' to replace the sensitive information, I wanted to complete this prior to obtaining the file on my home computer.

```
file_path <- "C:/Users/alyse/DIDPerformance.xlsx"
performance <- read_excel(file_path, sheet = "Performance")
benchmarks <- read_excel(file_path, sheet = "CMS Benchmarks 2022")
```

```
head(performance)
```

```
## # A tibble: 6 x 21
##   ‘Clinic Pseudonym‘ ‘Clinician Pseudonym‘ Role  ‘236_BP_NUM‘ ‘236_BP_DEN‘
##   <chr>              <chr>                 <chr>        <dbl>        <dbl>
## 1 Le Cordon Bleu     Allspice              MD             291          395
## 2 Le Cordon Bleu     Angelica              MD             210          274
## 3 Le Cordon Bleu     Anise                 PA              80          139
## 4 Le Cordon Bleu     Bay                   MD             211          263
## 5 Le Cordon Bleu     Basil                 NP              87          115
## 6 Le Cordon Bleu     Barberry              NP              85          110
## # i 16 more variables: ‘236_RATE‘ <dbl>, ‘112_BCS_NUM‘ <dbl>,
## #   ‘112_BCS_DEN‘ <dbl>, ‘112_RATE‘ <dbl>, ‘113_CRC_NUM‘ <dbl>,
## #   ‘113_CRC_DEN‘ <dbl>, ‘113_CRC_RATE‘ <dbl>, ‘309_CCS_NUM‘ <dbl>,
## #   ‘309_CCS_DEN‘ <dbl>, ‘309_CCS_RATE‘ <dbl>, ‘47_ACP_NUM‘ <dbl>,
## #   ‘47_ACP_DEN‘ <dbl>, ‘47_ACP_RATE‘ <dbl>, ‘134_DEP_NUM‘ <dbl>,
## #   ‘134_DEP_DEN‘ <dbl>, ‘134_DEP_RATE‘ <dbl>
```

```r
head(benchmarks)
```

```
## # A tibble: 6 x 19
##   ‘Measure Title‘  ‘Measure ID‘ ‘Collection Type‘ ‘Measure Type‘ ‘High Priority‘
##   <chr>            <chr>        <chr>             <chr>          <chr>
## 1 Controlling Hig~ 236          eCQM              Intermediate ~ Y
## 2 Breast Cancer S~ 112          eCQM              Process        N
## 3 Colorectal Canc~ 113          eCQM              Process        N
## 4 Cervical Cancer~ 309          eCQM              Process        N
## 5 Advance Care Pl~ 047          Medicare Part B ~ Process        Y
## 6 Preventive Care~ 134          eCQM              Process        N
## # i 14 more variables: ‘Average Performance Rate‘ <chr>,
## #   ‘Has a Benchmark‘ <chr>, ‘Decile 1‘ <chr>, ‘Decile 2‘ <chr>,
## #   ‘Decile 3‘ <chr>, ‘Decile 4‘ <chr>, ‘Decile 5‘ <chr>, ‘Decile 6‘ <chr>,
## #   ‘Decile 7‘ <chr>, ‘Decile 8‘ <chr>, ‘Decile 9‘ <chr>, ‘Decile 10‘ <chr>,
## #   ‘Topped Out?‘ <chr>, ‘7 Point Cap?‘ <chr>
```

Data Cleaning After checking to ensure the data was loaded and read correctly, I needed to check for missing values, as I know some of the clinicians are missing calculated rates, especially when the denominator is zero. I don't believe I need to rename any variables as the measure ID tends to follow the CMS naming convention, but that is another aspect I will check now that my data is present. I will also consider any data consolidation that may be needed utilizing joins and binds on key identifiers.

```r
na_counts_performance <- performance %>%
  summarize_all(~ sum(is.na(.)))
```

In this summary, I found one observation in the performance data frame that was empty across all 21 variables. I chose to apply a function that checks all rows again and if all elements are an empty string or NA the row would be removed.

```r
performance <- performance[!apply(performance, 1, function(x) all(x == "" | is.na(x))), ]
```

I completed the same action on the benchmark dataframe, and having the same findings, decided that the "empty row" is simply the bottom row of each worksheet and I don't believe it will impact my datasets at all.

```r
na_counts_benchmarks <- benchmarks %>%
  summarize_all(~ sum(is.na(.)))
```

Another thing I realized is that my performance variable titles for measure rates are not named consistently. Some have the measure ID_RATE, while others have included a shorthand to the measure title, like 113_CRC_RATE for the colorectal cancer screening measure. I renamed the rate columns for consistency and checked them after I ran the code.

```
performance <- performance %>%
  rename(
    `113_RATE` = `113_CRC_RATE`,
    `309_RATE` = `309_CCS_RATE`,
    `47_RATE` = `47_ACP_RATE`,
    `134_RATE` = `134_DEP_RATE`)
```

```
names(performance)
```

```
##  [1] "Clinic Pseudonym"    "Clinician Pseudonym" "Role"
##  [4] "236_BP_NUM"          "236_BP_DEN"          "236_RATE"
##  [7] "112_BCS_NUM"         "112_BCS_DEN"         "112_RATE"
## [10] "113_CRC_NUM"         "113_CRC_DEN"         "113_RATE"
## [13] "309_CCS_NUM"         "309_CCS_DEN"         "309_RATE"
## [16] "47_ACP_NUM"          "47_ACP_DEN"          "47_RATE"
## [19] "134_DEP_NUM"         "134_DEP_DEN"         "134_RATE"
```

Data Consolidation I reviewed the data and the questions I am interested in answering and found that in this project, I don't actually want to merge my data in any way, using joins or bind, although I will want to cross-reference the details of each measure later in both the performance and benchmark dataframes. Examining the data, however, made me realize that the benchmark ranges are percentages, rounded to the 2nd decimal place (example: 2.74 - 41.95 really means 2.74% - 41.95%). The performance rates, on the other hand, were structured as percentages in the original excel, but now reads as decimals (example: 0.7277628 should be expressed as 72.78%). I use mutate to make this change.

```
performance <- performance %>%
  mutate(across(ends_with("_RATE"), ~ round(.x * 100, 2)))
```

Condensed Display of Final Data Below are a few different views of my dataset:

```
str(performance)
```

```
## tibble [62 x 21] (S3: tbl_df/tbl/data.frame)
##  $ Clinic Pseudonym   : chr [1:62] "Le Cordon Bleu" "Le Cordon Bleu" "Le Cordon Bleu" "Le Cordon Bleu
##  $ Clinician Pseudonym: chr [1:62] "Allspice" "Angelica" "Anise" "Bay" ...
##  $ Role               : chr [1:62] "MD" "MD" "PA" "MD" ...
##  $ 236_BP_NUM         : num [1:62] 291 210 80 211 87 85 113 108 308 114 ...
##  $ 236_BP_DEN         : num [1:62] 395 274 139 263 115 110 162 142 352 160 ...
##  $ 236_RATE           : num [1:62] 73.7 76.6 57.5 80.2 75.7 ...
##  $ 112_BCS_NUM        : num [1:62] 76 37 22 313 135 149 199 166 119 93 ...
##  $ 112_BCS_DEN        : num [1:62] 100 57 48 419 185 232 269 219 151 128 ...
##  $ 112_RATE           : num [1:62] 76 64.9 45.8 74.7 73 ...
##  $ 113_CRC_NUM        : num [1:62] 683 374 152 497 217 161 278 235 416 143 ...
##  $ 113_CRC_DEN        : num [1:62] 849 527 252 695 310 316 452 335 508 200 ...
##  $ 113_RATE           : num [1:62] 80.5 71 60.3 71.5 70 ...
##  $ 309_CCS_NUM        : num [1:62] 174 93 92 502 531 346 363 355 113 71 ...
##  $ 309_CCS_DEN        : num [1:62] 198 118 148 658 623 478 466 466 152 96 ...
##  $ 309_RATE           : num [1:62] 87.9 78.8 62.2 76.3 85.2 ...
##  $ 47_ACP_NUM         : num [1:62] 270 126 48 222 64 60 125 119 257 135 ...
##  $ 47_ACP_DEN         : num [1:62] 371 227 104 314 83 91 198 169 316 171 ...
##  $ 47_RATE            : num [1:62] 72.8 55.5 46.1 70.7 77.1 ...
##  $ 134_DEP_NUM        : num [1:62] 1127 726 478 857 566 ...
##  $ 134_DEP_DEN        : num [1:62] 1365 940 673 1100 725 ...
##  $ 134_RATE           : num [1:62] 82.6 77.2 71 77.9 78.1 ...
```

I am really happy with the consistency and formatting I am seeing here.

```
summary(performance)
```

```
##  Clinic Pseudonym   Clinician Pseudonym     Role             236_BP_NUM
##  Length:62          Length:62           Length:62          Min.   :  1.0
##  Class :character   Class :character    Class :character   1st Qu.: 87.5
##  Mode  :character   Mode  :character    Mode  :character   Median :162.5
##                                                            Mean   :178.5
##                                                            3rd Qu.:245.0
##                                                            Max.   :444.0
##    236_BP_DEN       236_RATE      112_BCS_NUM     112_BCS_DEN
##  Min.   :  2.0   Min.   :48.28   Min.   :  0.0   Min.   :  1.0
##  1st Qu.:122.5   1st Qu.:68.20   1st Qu.: 74.5   1st Qu.:107.0
##  Median :231.0   Median :71.30   Median :115.5   Median :173.0
##  Mean   :249.2   Mean   :71.26   Mean   :166.7   Mean   :236.4
##  3rd Qu.:369.2   3rd Qu.:75.61   3rd Qu.:253.2   3rd Qu.:350.8
##  Max.   :599.0   Max.   :87.50   Max.   :501.0   Max.   :646.0
##     112_RATE       113_CRC_NUM     113_CRC_DEN        113_RATE
##  Min.   :  0.00   Min.   :  1.0   Min.   :   5.0   Min.   :20.00
##  1st Qu.: 64.17   1st Qu.:153.5   1st Qu.: 262.5   1st Qu.:59.27
##  Median : 68.24   Median :277.5   Median : 471.5   Median :66.73
##  Mean   : 66.72   Mean   :315.9   Mean   : 475.1   Mean   :64.14
##  3rd Qu.: 73.83   3rd Qu.:453.2   3rd Qu.: 684.8   3rd Qu.:71.51
##  Max.   :100.00   Max.   :777.0   Max.   :1054.0   Max.   :81.89
##   309_CCS_NUM     309_CCS_DEN       309_RATE        47_ACP_NUM
##  Min.   :  0.0   Min.   :  1.0   Min.   :  0.00   Min.   :  0.0
##  1st Qu.: 93.5   1st Qu.:148.2   1st Qu.: 63.77   1st Qu.: 61.0
##  Median :209.5   Median :316.0   Median : 73.46   Median :125.5
##  Mean   :245.9   Mean   :334.2   Mean   : 66.55   Mean   :173.7
##  3rd Qu.:378.2   3rd Qu.:513.0   3rd Qu.: 77.98   3rd Qu.:264.2
##  Max.   :750.0   Max.   :885.0   Max.   :100.00   Max.   :598.0
##    47_ACP_DEN       47_RATE       134_DEP_NUM      134_DEP_DEN
##  Min.   :  0.0   Min.   : 0.00   Min.   :   8.0   Min.   :  16.0
##  1st Qu.: 99.5   1st Qu.:59.59   1st Qu.: 383.0   1st Qu.: 515.2
##  Median :230.0   Median :69.64   Median : 573.5   Median : 728.5
##  Mean   :245.4   Mean   :66.75   Mean   : 570.1   Mean   : 721.9
##  3rd Qu.:362.0   3rd Qu.:78.49   3rd Qu.: 757.8   3rd Qu.: 947.5
##  Max.   :700.0   Max.   :95.65   Max.   :1191.0   Max.   :1407.0
##     134_RATE
##  Min.   :50.00
##  1st Qu.:77.00
##  Median :78.92
##  Mean   :78.28
##  3rd Qu.:82.12
##  Max.   :90.09
```

The median and mean of each variable, especially for performance rates will be important later. In addition, some of the measures, like depression screening (134) can give insight into the relative size of the patient population as the only requirement for inclusion in the denominator is to be at least 18 years old and have had one office visit in the performance year.

```
str(benchmarks)
```

```
## tibble [8 x 19] (S3: tbl_df/tbl/data.frame)
##  $ Measure Title          : chr [1:8] "Controlling High Blood Pressure" "Breast Cancer Screening" "C
##  $ Measure ID             : chr [1:8] "236" "112" "113" "309" ...
##  $ Collection Type        : chr [1:8] "eCQM" "eCQM" "eCQM" "eCQM" ...
##  $ Measure Type           : chr [1:8] "Intermediate Outcome" "Process" "Process" "Process" ...
##  $ High Priority          : chr [1:8] "Y" "N" "N" "N" ...
##  $ Average Performance Rate: chr [1:8] "62.30" "50.99" "49.64" "36.44" ...
##  $ Has a Benchmark        : chr [1:8] "Y" "Y" "Y" "Y" ...
##  $ Decile 1               : chr [1:8] "2.74 - 41.95" "0.27 - 9.22" "0.18 - 7.21" "0.44 - 7.76" ...
##  $ Decile 2               : chr [1:8] "41.96 - 51.35" "9.23 - 27.55" "7.22 - 22.60" "7.77 - 15.58"
##  $ Decile 3               : chr [1:8] "51.36 - 56.60" "27.56 - 39.41" "22.61 - 34.52" "15.59 - 21.87
##  $ Decile 4               : chr [1:8] "56.61 - 60.70" "39.42 - 48.17" "34.53 - 43.89" "21.88 - 27.95
##  $ Decile 5               : chr [1:8] "60.71 - 64.23" "48.18 - 54.83" "43.90 - 51.88" "27.96 - 34.03
##  $ Decile 6               : chr [1:8] "64.24 - 67.54" "54.84 - 60.56" "51.89 - 59.64" "34.04 - 40.16
##  $ Decile 7               : chr [1:8] "67.55 - 71.09" "60.57 - 66.81" "59.65 - 66.97" "40.17 - 46.98
##  $ Decile 8               : chr [1:8] "71.10 - 75.27" "66.82 - 73.30" "66.98 - 75.50" "46.99 - 54.54
##  $ Decile 9               : chr [1:8] "75.28 - 81.34" "73.31 - 82.04" "75.51 - 85.68" "54.55 - 68.51
##  $ Decile 10              : chr [1:8] ">= 81.35" ">= 82.05" ">= 85.69" ">= 68.52" ...
##  $ Topped Out?            : chr [1:8] "No" "No" "No" "No" ...
##  $ 7 Point Cap?           : chr [1:8] "No" "No" "No" "No" ...
```

I am happy with the structure of my benchmark df as well, and the percentage range is formatted in alignment with the rates in my performance df.

Learning Objectives and Knowledge Gaps

I don't think there are any obvious gaps in my knowledge of import and cleanup of the data. I have experience with both excel and comma delimited csv files, although the arff file this week was new to me.
Some of the information in this dataset that is not self evident include whether or not clinicians working at specific locations is a likely determinant of higher scores, or if MDs are more or less likely to score higher than their NP or PA counterparts. I am also hoping to find both low-performing and high-performing outliers as they may represent characteristics the practice managers may want to either discourage or replicate.

Exploratory Data Analysis (EDA) and Descriptive Statistics

Slice, Dice, and Statistical Exploration My first step of this section is to calculate and display some descriptive statistics (mean, median, range, etc.) for the performance metrics. As I explore the data further, I will want to plot and visualize the distribution of KPI in each dataset, and finally, explore patterns and potentially identify any outliers or anomalies.

Let's begin with some basic descriptive statistics for all clinicians and their performance rates.

```r
descriptive_stats <- performance %>%
  select(contains("_RATE")) %>%
  summarise_all(list(
    mean = ~mean(., na.rm = TRUE),
    median = ~median(., na.rm = TRUE),
    range_min = ~min(., na.rm = TRUE),
    range_max = ~max(., na.rm = TRUE),
    sd = ~sd(., na.rm = TRUE)))
print(descriptive_stats)
```

```
## # A tibble: 1 x 30
##   ‘236_RATE_mean‘ ‘112_RATE_mean‘ ‘113_RATE_mean‘ ‘309_RATE_mean‘ ‘47_RATE_mean‘
##             <dbl>           <dbl>           <dbl>           <dbl>          <dbl>
## 1            71.3            66.7            64.1            66.5           66.7
## # i 25 more variables: ‘134_RATE_mean‘ <dbl>, ‘236_RATE_median‘ <dbl>,
## #   ‘112_RATE_median‘ <dbl>, ‘113_RATE_median‘ <dbl>, ‘309_RATE_median‘ <dbl>,
## #   ‘47_RATE_median‘ <dbl>, ‘134_RATE_median‘ <dbl>,
## #   ‘236_RATE_range_min‘ <dbl>, ‘112_RATE_range_min‘ <dbl>,
## #   ‘113_RATE_range_min‘ <dbl>, ‘309_RATE_range_min‘ <dbl>,
## #   ‘47_RATE_range_min‘ <dbl>, ‘134_RATE_range_min‘ <dbl>,
## #   ‘236_RATE_range_max‘ <dbl>, ‘112_RATE_range_max‘ <dbl>, ...
```

The median and mean of each measure wasn’t overly surprising, nor were the upper/lower ranges. Seeing the standard deviation was more compelling, with a higher deviation potentially suggesting there could be opportunities for improvement and standardization of processes. In the measures with a lower deviation, performance is more consistent and closer to the mean, which could indicate well-established procedures and/or guidelines.

Before moving on to joins and creating some tables, which seems like the best way to quickly view some of this information, I would like to also calculate the mean, median, range, and standard deviation across only the MDs, then the NPs, then the PAs. Having these calculations based on all clinicians within each clinic will also likely prove useful.

```r
# For all MDs only
md_stats <- performance %>%
  filter(Role == "MD") %>%
  select(contains("_RATE")) %>%
  summarise_all(list(
    mean = ~mean(., na.rm = TRUE),
    median = ~median(., na.rm = TRUE),
    range_min = ~min(., na.rm = TRUE),
    range_max = ~max(., na.rm = TRUE),
    sd = ~sd(., na.rm = TRUE)))

# For all NPs only
np_stats <- performance %>%
  filter(Role == "NP") %>%
  select(contains("_RATE")) %>%
  summarise_all(list(
    mean = ~mean(., na.rm = TRUE),
    median = ~median(., na.rm = TRUE),
    range_min = ~min(., na.rm = TRUE),
    range_max = ~max(., na.rm = TRUE),
    sd = ~sd(., na.rm = TRUE)))

# For all PAs only
pa_stats <- performance %>%
  filter(Role == "PA") %>%
  select(contains("_RATE")) %>%
  summarise_all(list(
    mean = ~mean(., na.rm = TRUE),
    median = ~median(., na.rm = TRUE),
    range_min = ~min(., na.rm = TRUE),
    range_max = ~max(., na.rm = TRUE),
    sd = ~sd(., na.rm = TRUE)))

# For all clinicians within each clinic
clinic_stats <- performance %>%
  group_by(`Clinic Pseudonym`) %>%
  select(contains("_RATE")) %>%
  summarise_all(list(
    mean = ~mean(., na.rm = TRUE),
    median = ~median(., na.rm = TRUE),
    range_min = ~min(., na.rm = TRUE),
    range_max = ~max(., na.rm = TRUE),
    sd = ~sd(., na.rm = TRUE)), .groups = "drop") # here I am trying to prevent the creation of an extr
```

```
## Adding missing grouping variables: ‘Clinic Pseudonym‘
```

Now that those are all created, I print the results to check them in each dataframe.

```r
print(descriptive_stats)
```

```
## # A tibble: 1 x 30
##   ‘236_RATE_mean‘ ‘112_RATE_mean‘ ‘113_RATE_mean‘ ‘309_RATE_mean‘ ‘47_RATE_mean‘
##             <dbl>           <dbl>           <dbl>           <dbl>          <dbl>
## 1            71.3            66.7            64.1            66.5           66.7
## # i 25 more variables: ‘134_RATE_mean‘ <dbl>, ‘236_RATE_median‘ <dbl>,
## #   ‘112_RATE_median‘ <dbl>, ‘113_RATE_median‘ <dbl>, ‘309_RATE_median‘ <dbl>,
## #   ‘47_RATE_median‘ <dbl>, ‘134_RATE_median‘ <dbl>,
## #   ‘236_RATE_range_min‘ <dbl>, ‘112_RATE_range_min‘ <dbl>,
## #   ‘113_RATE_range_min‘ <dbl>, ‘309_RATE_range_min‘ <dbl>,
## #   ‘47_RATE_range_min‘ <dbl>, ‘134_RATE_range_min‘ <dbl>,
## #   ‘236_RATE_range_max‘ <dbl>, ‘112_RATE_range_max‘ <dbl>, ...
```

```r
print(md_stats)
```

```
## # A tibble: 1 x 30
##   ‘236_RATE_mean‘ ‘112_RATE_mean‘ ‘113_RATE_mean‘ ‘309_RATE_mean‘ ‘47_RATE_mean‘
##             <dbl>           <dbl>           <dbl>           <dbl>          <dbl>
## 1            72.0            68.2            66.7            68.1           70.8
## # i 25 more variables: ‘134_RATE_mean‘ <dbl>, ‘236_RATE_median‘ <dbl>,
## #   ‘112_RATE_median‘ <dbl>, ‘113_RATE_median‘ <dbl>, ‘309_RATE_median‘ <dbl>,
## #   ‘47_RATE_median‘ <dbl>, ‘134_RATE_median‘ <dbl>,
## #   ‘236_RATE_range_min‘ <dbl>, ‘112_RATE_range_min‘ <dbl>,
## #   ‘113_RATE_range_min‘ <dbl>, ‘309_RATE_range_min‘ <dbl>,
## #   ‘47_RATE_range_min‘ <dbl>, ‘134_RATE_range_min‘ <dbl>,
## #   ‘236_RATE_range_max‘ <dbl>, ‘112_RATE_range_max‘ <dbl>, ...
```

```
print(np_stats)
```

```
## # A tibble: 1 x 30
##   ‘236_RATE_mean‘ ‘112_RATE_mean‘ ‘113_RATE_mean‘ ‘309_RATE_mean‘ ‘47_RATE_mean‘
##             <dbl>           <dbl>           <dbl>           <dbl>          <dbl>
## 1            68.8            63.2            57.0            59.8           58.2
## # i 25 more variables: ‘134_RATE_mean‘ <dbl>, ‘236_RATE_median‘ <dbl>,
## #   ‘112_RATE_median‘ <dbl>, ‘113_RATE_median‘ <dbl>, ‘309_RATE_median‘ <dbl>,
## #   ‘47_RATE_median‘ <dbl>, ‘134_RATE_median‘ <dbl>,
## #   ‘236_RATE_range_min‘ <dbl>, ‘112_RATE_range_min‘ <dbl>,
## #   ‘113_RATE_range_min‘ <dbl>, ‘309_RATE_range_min‘ <dbl>,
## #   ‘47_RATE_range_min‘ <dbl>, ‘134_RATE_range_min‘ <dbl>,
## #   ‘236_RATE_range_max‘ <dbl>, ‘112_RATE_range_max‘ <dbl>, ...
```

```
print(pa_stats)
```

```
## # A tibble: 1 x 30
##   ‘236_RATE_mean‘ ‘112_RATE_mean‘ ‘113_RATE_mean‘ ‘309_RATE_mean‘ ‘47_RATE_mean‘
##             <dbl>           <dbl>           <dbl>           <dbl>          <dbl>
## 1            71.3            62.9            60.5            70.9           53.6
## # i 25 more variables: ‘134_RATE_mean‘ <dbl>, ‘236_RATE_median‘ <dbl>,
## #   ‘112_RATE_median‘ <dbl>, ‘113_RATE_median‘ <dbl>, ‘309_RATE_median‘ <dbl>,
## #   ‘47_RATE_median‘ <dbl>, ‘134_RATE_median‘ <dbl>,
## #   ‘236_RATE_range_min‘ <dbl>, ‘112_RATE_range_min‘ <dbl>,
## #   ‘113_RATE_range_min‘ <dbl>, ‘309_RATE_range_min‘ <dbl>,
## #   ‘47_RATE_range_min‘ <dbl>, ‘134_RATE_range_min‘ <dbl>,
## #   ‘236_RATE_range_max‘ <dbl>, ‘112_RATE_range_max‘ <dbl>, ...
```

```
print(clinic_stats)
```

```
## # A tibble: 8 x 31
##   ‘Clinic Pseudonym‘    ‘236_RATE_mean‘ ‘112_RATE_mean‘ ‘113_RATE_mean‘
##   <chr>                           <dbl>           <dbl>           <dbl>
## 1 Boston University                70.3            66.2            70.3
## 2 Culinary Arts Academy            63.6            63.1            56.9
## 3 Culinary Institute               74.2            70.5            70.3
## 4 Escoffier School                 74.6            69.8            64.9
## 5 Hattori Nutrition                66.8            62.8            55.7
## 6 Kendall College                  67.2            58.2            56.1
## 7 La Cuisine Paris                 73.9            67.8            63.5
## 8 Le Cordon Bleu                   73.4            68.6            67.0
## # i 27 more variables: ‘309_RATE_mean‘ <dbl>, ‘47_RATE_mean‘ <dbl>,
## #   ‘134_RATE_mean‘ <dbl>, ‘236_RATE_median‘ <dbl>, ‘112_RATE_median‘ <dbl>,
## #   ‘113_RATE_median‘ <dbl>, ‘309_RATE_median‘ <dbl>, ‘47_RATE_median‘ <dbl>,
## #   ‘134_RATE_median‘ <dbl>, ‘236_RATE_range_min‘ <dbl>,
## #   ‘112_RATE_range_min‘ <dbl>, ‘113_RATE_range_min‘ <dbl>,
## #   ‘309_RATE_range_min‘ <dbl>, ‘47_RATE_range_min‘ <dbl>,
## #   ‘134_RATE_range_min‘ <dbl>, ‘236_RATE_range_max‘ <dbl>, ...
```

I would like to display these in a small table along with measure IDs and measure names, which exist in the benchmark dataframe. Because I know I will be joining a lot on the Measure ID, I need to convert all of my newly created "stats" dataframes into a long format. This will help me insure that Measure ID is a numeric field across all of them, making for fewer errors as I write my joins.

```r
# First, I want to create the function.
to_long_format <- function(df) {
  df %>%
    pivot_longer(cols = -Role, names_to = "measure_var", values_to = "value") %>%
    separate(measure_var, into = c("measure_id", "stat"), sep = "_RATE_") %>%
    mutate(measure_id = as.numeric(measure_id)) %>%
    pivot_wider(names_from = stat, values_from = value)}
```

Upon examination of the stats dataframes, I want to be more careful with the clinic_stats df because it has 8 observations instead of just the 1 that the others have. I also need to add a "role" column to the stats dataframes because the results in each was filtered by role. While there may be a simpler solution, the easiest way I can think of to solve for this is to add a constant "Role" column with the appropriate role filled in as values all the way down.

```r
md_stats$Role <- "MD"
np_stats$Role <- "NP"
pa_stats$Role <- "PA"
```

Now I think I am ready to apply the long format function to the first three.

```r
# Applying the function to each statistics dataframe
md_stats_long <- to_long_format(md_stats)
np_stats_long <- to_long_format(np_stats)
pa_stats_long <- to_long_format(pa_stats)
```

That worked well, but I think I would like to have a separate function just for the clinic conversion to long format.

```r
to_long_format_clinic <- function(df) {
  df %>%
    pivot_longer(cols = -c('Clinic Pseudonym'), names_to = "measure_var", values_to = "value") %>%
    separate(measure_var, into = c("measure_id", "stat"), sep = "_RATE_") %>%
    mutate(measure_id = as.numeric(measure_id)) %>% # Converting measure_id to numeric
    pivot_wider(names_from = stat, values_from = value)} # Not sure which is better - longer vs wider
```

Now I am ready to apply this function to the clinic_stats dataframe.

```r
clinic_stats_long <- to_long_format_clinic(clinic_stats)
```

Finally, I would like to convert the 'descriptive_stats' dataframe to long form as it includes all clinicians from all clinics, so is the best "rollup" for a high-level organizational view.

```r
to_long_format_general <- function(df) {
  df %>%
    pivot_longer(cols = everything(), names_to = "measure_var", values_to = "value") %>%
    separate(measure_var, into = c("measure_id", "stat"), sep = "_RATE_") %>%
    mutate(measure_id = as.numeric(measure_id)) %>%
    pivot_wider(names_from = stat, values_from = value)}
```

And here is the application of the function for the organizational stats.

```
descriptive_stats_long <- to_long_format_general(descriptive_stats)
```

Although this was time consuming, I think it will serve me better in the long run, especially as I start to explore tables and visualizations. Here are my long format dataframes.

```
print(descriptive_stats_long)
```

```
## # A tibble: 6 x 6
##   measure_id  mean median range_min range_max    sd
##        <dbl> <dbl>  <dbl>     <dbl>     <dbl> <dbl>
## 1        236  71.3   71.3      48.3      87.5  7.27
## 2        112  66.7   68.2       0       100   13.3
## 3        113  64.1   66.7      20        81.9 11.9
## 4        309  66.5   73.5       0       100   20.9
## 5         47  66.7   69.6       0        95.6 18.3
## 6        134  78.3   78.9      50        90.1  6.34
```

```
print(md_stats_long)
```

```
## # A tibble: 6 x 7
##   Role  measure_id  mean median range_min range_max    sd
##   <chr>      <dbl> <dbl>  <dbl>     <dbl>     <dbl> <dbl>
## 1 MD           236  72.0   71.3      54.3      87.5  6.50
## 2 MD           112  68.2   71.0      40        84.3  9.94
## 3 MD           113  66.7   69.7      43.6      81.9  9.76
## 4 MD           309  68.1   74.8      19.0     100   19.4
## 5 MD            47  70.8   72.8      18.2      95.6 14.6
## 6 MD           134  79.2   79.3      62.0      90.1  5.36
```

```
print(np_stats_long)
```

```
## # A tibble: 6 x 7
##   Role  measure_id  mean median range_min range_max    sd
##   <chr>      <dbl> <dbl>  <dbl>     <dbl>     <dbl> <dbl>
## 1 NP           236  68.8   70        48.3      77.3  9.31
## 2 NP           112  63.2   66.9       0       100   22.0
## 3 NP           113  57.0   61.8      20        80   17.3
## 4 NP           309  59.8   72.4       0        85.2 28.2
## 5 NP            47  58.2   66.7       0        86.6 26.3
## 6 NP           134  76.8   78.1      50        86.4  8.97
```

```
print(pa_stats_long)
```

```
## # A tibble: 6 x 7
##   Role  measure_id  mean median range_min range_max    sd
##   <chr>      <dbl> <dbl>  <dbl>     <dbl>     <dbl> <dbl>
## 1 PA           236  71.3   73.6      57.6      79.0  8.19
## 2 PA           112  62.9   65.7      45.8      70.6  9.79
## 3 PA           113  60.5   60.3      57.7      64.1  2.64
## 4 PA           309  70.9   72.1      62.2      75.8  5.21
## 5 PA            47  53.6   55.7      40.7      67.3 10.4
## 6 PA           134  74.4   73.5      68.0      80.5  5.30
```

```
print(clinic_stats_long)
```

```
## # A tibble: 48 x 7
##    ‘Clinic Pseudonym‘    measure_id  mean median range_min range_max    sd
##    <chr>                      <dbl> <dbl>  <dbl>     <dbl>     <dbl> <dbl>
##  1 Boston University            236  70.3   69.4      63.7      80.3  5.12
##  2 Boston University            112  66.2   67.2      45.8      80.6 10.3
##  3 Boston University            113  70.3   70        59.1      79.9  6.85
##  4 Boston University            309  72.0   72.7      33.0      89.5 16.7
##  5 Boston University             47  75.5   73.0      56.9      95.6 13.9
##  6 Boston University            134  80.6   80.6      74.3      85.0  3.70
##  7 Culinary Arts Academy        236  63.6   67.8      48.3      75   11.8
##  8 Culinary Arts Academy        112  63.1   70.6       0       100   33.9
##  9 Culinary Arts Academy        113  56.9   68.1      20        80   23.5
## 10 Culinary Arts Academy        309  50.0   69.6       0        84.0 39.3
## # i 38 more rows
```

Summarizing early patterns

The data has all been sliced and diced appropriately and prepared for my next step of creating data visualizations and joining with the benchmark data to easily bring over measure titles and national performance data ranges. I will want to use some joined tables and basic comparisons and/or visualizations, like histograms or heat maps to display some of the early patterns I am finding in the data.

One example of this includes comparing the mean of each measure across all three roles.

```
# Selecting only measure_id and mean from each dataframe and renaming the mean column to reflect the ro
md_means <- md_stats_long %>% select(measure_id, MD_mean = mean)
np_means <- np_stats_long %>% select(measure_id, NP_mean = mean)
pa_means <- pa_stats_long %>% select(measure_id, PA_mean = mean)

# Joining the dataframes by measure_id
role_means <- full_join(md_means, np_means, by = "measure_id") %>%
              full_join(pa_means, by = "measure_id")
```

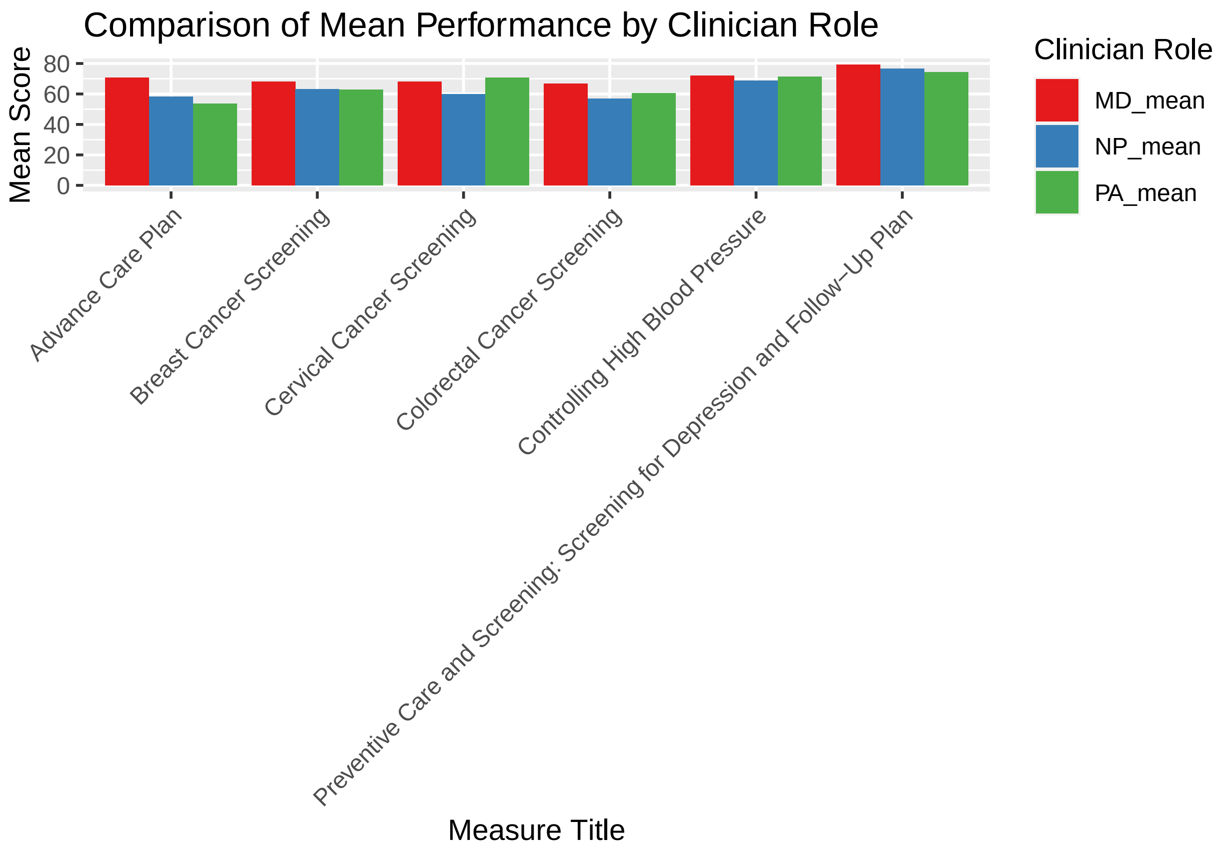
```
# Making sure that 'Measure ID' in the benchmarks dataframe is numeric
benchmarks <- benchmarks %>%
  mutate(`Measure ID` = as.numeric(`Measure ID`))

# Joining with the benchmarks dataframe to get measure titles
role_comparison_with_titles <- role_means %>%
  left_join(select(benchmarks, `Measure ID`, `Measure Title`), by = c("measure_id" = "Measure ID"))

# Arranging by measure_id for easier reading
role_comparison_with_titles <- role_comparison_with_titles %>%
  arrange(measure_id) %>%
  select(measure_id, `Measure Title`, MD_mean, NP_mean, PA_mean)
```

```r
# View the table
print(role_comparison_with_titles)
```

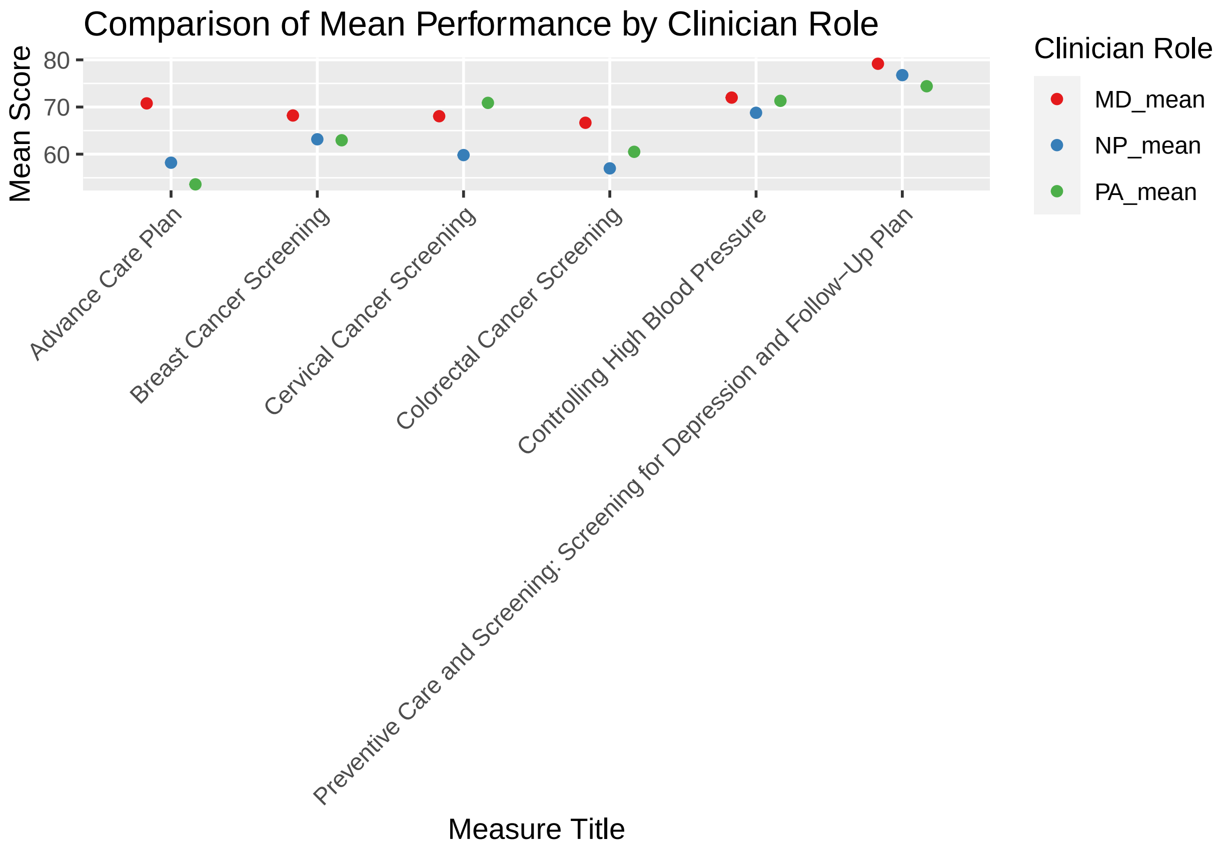
```
## # A tibble: 6 x 5
##   measure_id `Measure Title`                                MD_mean NP_mean PA_mean
##        <dbl> <chr>                                            <dbl>   <dbl>   <dbl>
## 1         47 Advance Care Plan                                 70.8    58.2    53.6
## 2        112 Breast Cancer Screening                           68.2    63.2    62.9
## 3        113 Colorectal Cancer Screening                       66.7    57.0    60.5
## 4        134 Preventive Care and Screening: Screening f~       79.2    76.8    74.4
## 5        236 Controlling High Blood Pressure                   72.0    68.8    71.3
## 6        309 Cervical Cancer Screening                         68.1    59.8    70.9
```

Two ways to look at this comparison visually would be with a grouped bar chart or a dot plot.

```r
# Converting the data to long format for plotting with ggplot2
long_role_comparison <- role_comparison_with_titles %>%
  gather(key = "Role", value = "Mean", MD_mean, NP_mean, PA_mean)

ggplot(long_role_comparison, aes(x = `Measure Title`, y = Mean, fill = Role)) +
  geom_bar(stat = "identity", position = position_dodge()) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(y = "Mean Score", x = "Measure Title", fill = "Clinician Role") +
  ggtitle("Comparison of Mean Performance by Clinician Role") +
  scale_fill_brewer(palette = "Set1")
```



```r
ggplot(long_role_comparison, aes(x = `Measure Title`, y = Mean, color = Role)) +
  geom_point(position = position_dodge(width = 0.5)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(y = "Mean Score", x = "Measure Title", color = "Clinician Role") +
  ggtitle("Comparison of Mean Performance by Clinician Role") +
  scale_color_brewer(palette = "Set1")
```



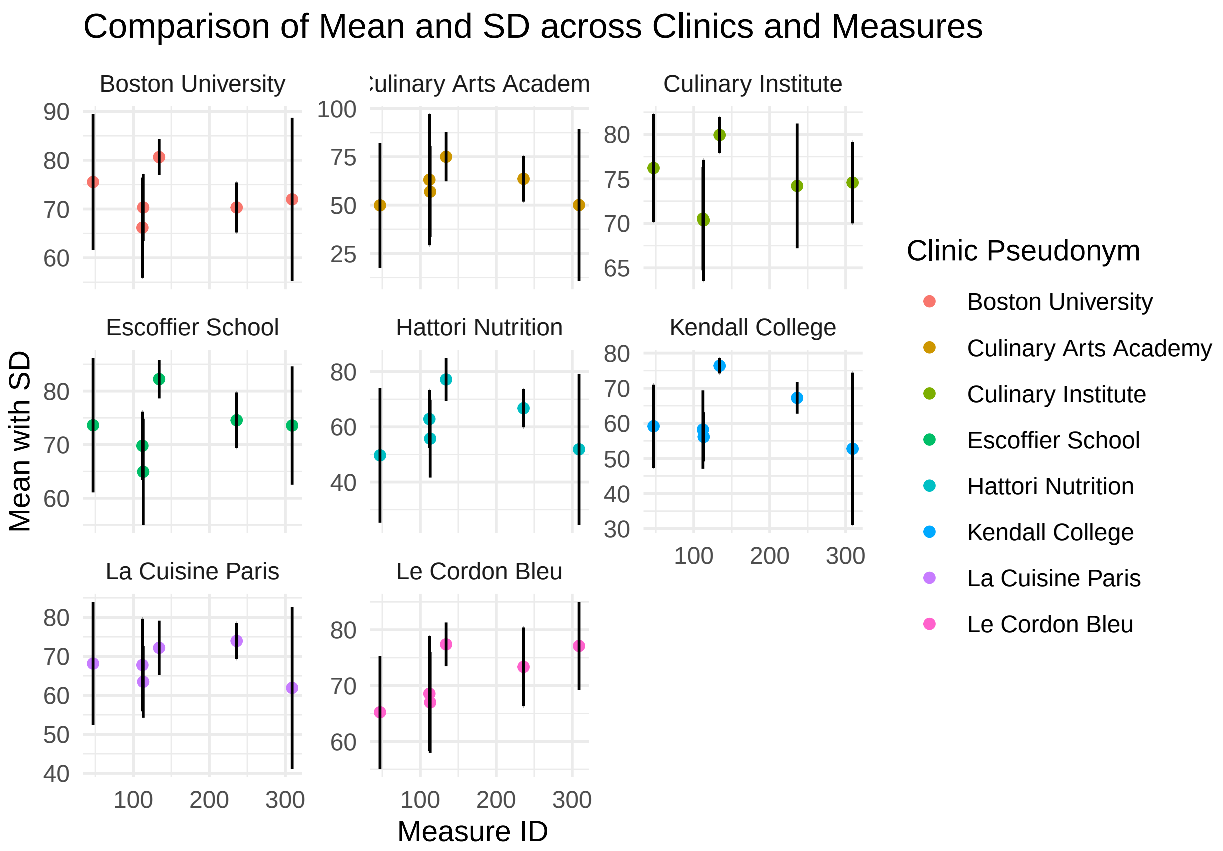
In addition to comparing performance by role, we can also look at all-provider performance segmented by clinic.

```r
# Ensuring that the 'Measure ID' columns in both dataframes have the same type
benchmarks <- benchmarks %>%
  mutate(`Measure ID` = as.numeric(as.character(`Measure ID`)))

clinic_stats_long <- clinic_stats_long %>%
  mutate(measure_id = as.numeric(as.character(measure_id)))

# Joining the clinic_stats_long dataframe with benchmarks to add measure titles
clinic_stats_with_titles <- clinic_stats_long %>%
  left_join(benchmarks, by = c("measure_id" = "Measure ID"))

# Creating a summary table with the required information
clinic_summary <- clinic_stats_with_titles %>%
  group_by(`Clinic Pseudonym`, measure_id, `Measure Title`) %>%
  summarize(
    Total_Mean = mean(mean, na.rm = TRUE),
    Total_SD = mean(sd, na.rm = TRUE),
    .groups = 'drop'  # This drops the grouping structure afterwards
    )

# Viewing the summary table
print(clinic_summary)
```

```
## # A tibble: 48 x 5
##    ‘Clinic Pseudonym‘    measure_id ‘Measure Title‘          Total_Mean Total_SD
##    <chr>                      <dbl> <chr>                         <dbl>    <dbl>
##  1 Boston University             47 Advance Care Plan              75.5    13.9
##  2 Boston University            112 Breast Cancer Screening        66.2    10.3
##  3 Boston University            113 Colorectal Cancer Scree~       70.3     6.85
##  4 Boston University            134 Preventive Care and Scr~       80.6     3.70
##  5 Boston University            236 Controlling High Blood ~       70.3     5.12
##  6 Boston University            309 Cervical Cancer Screeni~       72.0    16.7
##  7 Culinary Arts Academy         47 Advance Care Plan              49.9    32.3
##  8 Culinary Arts Academy        112 Breast Cancer Screening        63.1    33.9
##  9 Culinary Arts Academy        113 Colorectal Cancer Scree~       56.9    23.5
## 10 Culinary Arts Academy        134 Preventive Care and Scr~       75.0    12.7
## # i 38 more rows
```

It may be too complex to show both mean performance and standard deviation across 8 clinics and 6 measures, but here is my attempt at a faceted error bar chart from the summary table:

```
library(ggplot2)

# Creating a faceted error bar plot
ggplot(clinic_summary, aes(x = measure_id, y = Total_Mean, group = `Clinic Pseudonym`)) +
  geom_point(aes(color = `Clinic Pseudonym`)) +  # Points for mean
  geom_errorbar(aes(ymin = Total_Mean - Total_SD, ymax = Total_Mean + Total_SD), width = 0.2) +  # Erro
  facet_wrap(~ `Clinic Pseudonym`, scales = 'free_y') +  # Create a facet for each clinic
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +  # Rotate x labels for better readability
  labs(x = "Measure ID", y = "Mean with SD", title = "Comparison of Mean and SD across Clinics and Measu
  theme_minimal()
```



##Some Commentary on Data Visualization I have begun the process of creating dataframes and tables from which I can create some strong data visualizations. My plots, honestly, need some work so far. I'm putting this note here to remind myself (and maybe you) that this is where I am leaving off for my step 2 of my final project. I have some ideas about what I want the more advanced analytics to display, but I really need to ensure my basic ones are first more readable and easier to cross-reference.

This exercise has already taught me valuable things about my dataset and the next time I work with it, I plan to do some additional cleanup and arrangement of dataframes a little differently than I have here this week.

Summarizing early patterns

Data Integration, Comparison, and Pattern Recognition

Once the comparative visualizations have been summarized and explained, I plan to dig a little deeper into the patterns and compare the distributions of KPI in the internal data to the national benchmarks, using correlation tests to identify any significant relationships. I anticipate scatter plots and box plots will help me best visualize the comparison within the data. In addition to the above, I could also apply unsupervised learning algorithms like k-means clustering to identify patterns or groupings of clinician performance that aren't easily identifiable. Interpreting these clusters will allow me to identify common characteristics or trends within each group.

Regression Analysis

This step involves using regression analyses to identify predictors of clinician performance. This could be a simple linear regression, or, if I can find multiple variables within the dataset that could function as predictors of performance, it may be a multiple regression. Although the performance results are not currently binary, I could add a binary variable which reflects whether or not a given clinician is meeting the internal target goal for this year and then apply logistic regression to model those outcomes. This approach needs further thought and better understanding on my end, however. Finally, for this step I will want to evaluate the fit of my regression models using R-squared, P-values, and residual plots.

Classification and Prediction

This step is dependent on whether I decide to create a binary variable related to meeting the current year's target goals. If that turns out to be a viable and relevant option, it may then prove useful to explore the creation of a logistic regression classifier. Assessing the accuracy of this classifier will also be important, so I would want to explore the testing metrics available for this. Regardless of the binary variable, I can use the k-nearest neighbors algorithm to create a model that predicts clinician performance in some capacity. I can use my regression models from the step above to refine my predictions for the upcoming year and validate the overall model to the best of my ability. Some of these steps will require a deeper dive into concepts covered in this term as I want to make sure I have the best understanding possible when applying it to this real-world problem.

Reporting and Recommendations

I will compile all of my findings, plots, and analyses in a cohesive report using R Markdown. This should provide clear and concise interpretations of my results. Based on these analyses, I can make recommendations for target goals and interventions to improve clinician performance. Ideally, I can also save my predictions of clinician performance and follow up at the end of the year to compare them to actuals.

How My Approach Addresses the Problem:

This is a comprehensive approach where I am trying to apply various statistical and machine learning techniques from this term to better understand and improve clinician quality of care. By comparing internal performance to national performance, I should be able to benchmark performance and identify areas for improvement.

The Center for Medicare and Medicaid (CMS) also provides benchmarking data, to which I can compare my own developed benchmarks to. In addition, the regression models, classification, and clustering should help me uncover patterns and predictors of performance, guiding some targeted interventions. Finally, the detailed report should provide clear and actionable insights for next steps in driving quality of care forward.

Data:

I originally obtained three datasets for the purposes of this project:

1. PY 2021 Clinician Public Reporting: MIPS Measures and Attestations -

The first set of data is from the Centers for Medicare and Medicaid (CMS) database and has a list of all individually reporting clinicians and details regarding each one's 2021 participation in the Merit-based Incentive System (MIPS) program, which is a value-based program that tracks and reimburses eligible clinician and clinician groups' efforts toward greater quality of care, lower cost of care, and higher patient satisfaction.

The URL for this dataset is: https://data.cms.gov/provider-data/dataset/7d6a-e7a6

2. 2021 Accountable Care Organization Financial and Quality Results -

The second set of data is also from the CMS database and has a list of all the registered accountable care organizations (ACO) and details regarding each one's 2021 participation in the Medicare Shared Savings Program (MSSP), which is a value-based program that tracks and reimburses ACOs efforts toward greater quality of care, lower cost of care, and higher patient satisfaction.

The URL for this dataset is: https://data.cms.gov/medicare-shared-savings-program/performance-year-financial-and-quality-results/data/january-2021

3. Internal De-Identified Clinician Quality Results -

The third set of data is a spreadsheet I've compiled of providers and practices along with their KPI results across six CMS-based measures, which are also represented either in part or whole in the other two datasets. This spreadsheet is specific to the work I do and a consulting engagement I have with a small health system. Due to the sensitive nature of this customer data, the clinician names and practice locations have been de-identified and I have exchanged provider names with the names of spices and clinic practice locations with the names of culinary schools.

Additional Data -

4. Streamlined CMS 2022 Benchmarking Data -

The fourth dataset is a simple spreadsheet which includes CMS-established national benchmarks for 2022 specific to the measures listed in the Internal De-Identified Clinician Quality Results spreadsheet. These are also found on cms.gov and are published annually to their QPP online library.

Required Packages:

There are a variety of packages I have explored so far in this class and there are a few more I have yet to use, but that I suspect could be useful within the scope of this final project. Here is a list of R packages I'm considering based on my approach:

Data Import and Manipulation

1. readr: for importing data from CSV files.
2. readxl: for reading data from Excel files.
3. tidyverse: a collection of packages for data manipulation and visualization which includes some of what is here and more.
4. dplyr: for data manipulation and transformation.
5. tidyr: for cleaning and reshaping data.

Data Visualization and Clustering

1. ggplot2: for creating static graphics.
2. plotly: for if I get really assertive and try to create an interactive plot.
3. corrplot: for visualizing correlation matrices.
4. cluster: provides functions for cluster analysis.

Statistical Analyses

1. caret: for creating and evaluating classification and regression models.
2. nnet: for fitting multinomial log-linear models, which can be used for logistic regression.
3. stats: base R package with functions for multiple statistical tests.

Machine Learning

1. class: for k-nearest neighbors algorithm
2. e1071: for creating confusion matrices.

Reporting

1. rmarkdown: for creating dynamic reports with R Markdown.
2. knitr: for turning analyses into high quality documents, reports, and presentations.

Plots and Table Needs:

Some of the plots and tables I may need to use include: * Histogram * Box Plot * Bar Chart * Missing Value Heatmap * Summary Statistics Table * Table of Frequency * Scatter Plot (with/without regression lines) * Correlation Matrix * ROC Curve (if I use a logistic regression classifier) * Confusion Matrix * Table of Model Performance

Questions for Future Steps:

One question which has arisen recently is if I could obtain performance on these same measures from the same clinicians over the prior two or three years. It is possible that past performance could prove to be a strong predictor of future ability to perform, which could strongly impact target goals any machine learning could recommend.

Final Thoughts:

I will continue to examine the data and work on visualizations while progressing through further coding of my sections. I want to try to keep this project as simple as possible while still working toward a useful output. I am sure it will change and develop between now and the final submission of my project. Any feedback or thoughts are welcomed!