Nurse staffing strategies for enhanced patient care

Matthew Bain

2024-03-22

abstract I analyze a medical staffing dataset and identify avenues to improve work satisfaction among nurses and the quality of care provided at United States medical institutions.

Table of contents

[...]

Imports

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from scipy import stats
from great_tables import GT
from itertools import combinations
from pandas.plotting import scatter_matrix
from IPython.display import (
    display as display3,
    Markdown
)

from src.stylesheet import customize_plots
from src.inspection import make_df, display, display2
```

The dataset

Load the data

We begin by exploring the data to get to know the features and patterns on which we will base our analysis.

```
if 'data' not in locals():
    data = pd.read_csv(
        "../data/raw/PBJ_Daily_Nurse_Staffing_Q1_2024.zip",
        encoding='ISO-8859-1',
        low_memory=False
    )
else:
    print("data loaded.")
```

Inspect the data

```
GT(data.sample(10))
```

PROVNUM	PROVNAME	CITY	COUNTY_NAME	Other columns (overlapping, illegible)
155655	PEABODY RETIREMENT COMMUNITY	NORTH MANCHESTER	[illegible]	[illegible]
245529	BIGFORK VALLEY COMMUNITIES	BIGFORK	[illegible]	[illegible]
435132	AURORA BRULE NURSING HOME INC	WHITE LAKE	Aurora	[illegible]
225404	ASPEN HILL REHABILITATION [illegible]	HAVERHILL	Essex	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

```python
df = data.describe().round(1)
GT(df.reset_index())
```

index	COUNTY_FIPS	WorkDate	MDScensus	Hrs_RNDON	Hrs_RNDON_emp	Hrs_RNDON_ctr	Hrs_RNadmin	Hrs_RNadmin_emp	Hrs_RNadmin_ctr	Hrs_RN	Hrs_RN_emp	Hrs_RN_ctr	Hrs_LPNadmin	Hrs_LPNadmin_emp	Hrs_LPNadmin_ctr	Hrs_LPN	Hrs_LPN_emp	Hrs_LPN_ctr	Hrs_CNA	Hrs_CNA_emp	Hrs_CNA_ctr	Hrs_NAtrn	Hrs_NAtrn_emp	Hrs_NAtrn_ctr	Hrs_MedAide	Hrs_MedAide_emp	Hrs_MedAide_ctr
count	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0	1330966.0
mean	[illegible]	[illegible]	83.4	5.2	5.1	0.1	10.3	10.0	0.2	34.4	31.5	3.0	6.6	6.6	0.1	66.3	59.8	6.5	171.2	158.2	13.0	4.2	4.2	0.1	8.5	8.3	0.2
std	[illegible]	[illegible]	49.1	4.5	4.5	0.9	14.9	14.6	1.8	34.7	31.4	10.7	10.7	10.6	1.3	48.4	44.8	[illegible]	[illegible]	106.3	32.6	13.1	12.7	2.1	17.6	17.2	2.2
min	[illegible]	[illegible]	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
25%	[illegible]	[illegible]	51.0	0.0	0.0	0.0	0.0	0.0	0.0	13.0	12.0	0.0	0.0	0.0	0.0	32.8	28.2	0.0	97.0	88.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
50%	[illegible]	[illegible]	76.0	8.0	8.0	0.0	7.5	7.4	0.0	25.6	24.2	0.0	0.0	0.0	0.0	56.9	50.8	[illegible]	[illegible]	136.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0
75%	[illegible]	[illegible]	104.0	8.0	8.0	0.0	16.0	16.0	0.0	44.8	41.5	0.0	9.0	8.8	0.0	88.6	81.2	[illegible]	[illegible]	203.1	11.0	0.0	0.0	0.0	11.2	10.8	0.0
max	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Group the features

We note that there are 91 records per provider (`len(data["WorkDate"].unique())`) and 1,330,966 records in the table overall. The following table, which collapses the raw data across providers, thus has 14,626 $\left(\frac{1330966}{91}\right)$ entries.

```python
df = (
    data.loc[:, [
        "STATE",
        "COUNTY_NAME", "COUNTY_FIPS",
        "CITY",
        "PROVNAME", "PROVNUM",
    ]]
    .value_counts()
    .to_frame()
    .rename(columns={0: 'Counts'})
)
GT(df.reset_index().head(n=20))
```

STATE	COUN-TY_NAME	COUN-TY_FIPS	CITY	PROV-NAME	PROVNUM	Counts
AK	Anchorage	20	ANCHOR-AGE	PRESTIGE CARE & REHAB CENTER OF AN-CHORAGE	025025	91
OH	Allen	3	LIMA	LIMA CONVA-LESCENT HOME	366297	91
OH	Allen	3	LIMA	SHAWNEE MANOR	365361	91
OH	Allen	3	LIMA	SPRINGS OF LIMA THE	366464	91
OH	Allen	3	LIMA	SPRINGVIEW MANOR	366221	91
OH	Allen	3	SPENCERVILLE	ROSELAWN MANOR	365744	91
OH	Ashland	5	ASHLAND	BRETHREN CARE VILLAGE HEALTH CARE CENTER	366166	91
OH	Ashland	5	ASHLAND	CRYS-TAL CARE CENTER OF ASH-LAND	366239	91
OH	Ashland	5	ASHLAND	GOOD SAMA-RITAN [illegible]	365093	91
OH	Ashland	5	ASHLAND [illegible]	[illegible]	365646	91
OH	**Ashtabula**	**7**	[illegible]	[illegible]	[illegible]	**91**

Table 1: Record counts across state, county and provider.

```
GT(data[["CY_Qtr", "WorkDate", "MDScensus"]].head())
```

CY_Qtr	WorkDate	MDScensus
2024Q1	20240101	50
2024Q1	20240102	49
2024Q1	20240103	49
2024Q1	20240104	50
2024Q1	20240105	51

Clean the data

Explore the dataset

Visualize distributions

Visualize relationships

```
attributes = ["Hrs_RN", "Hrs_LPN_ctr", "Hrs_CNA", "Hrs_NAtrn", "Hrs_MedAide"]
n = len(attributes)

fig, axs = plt.subplots(n, n, figsize=(8, 8))
scatter_matrix(
    data[attributes].sample(200),
    ax=axs, alpha=.7,
    hist_kwds=dict(bins=15, linewidth=0)
)
fig.align_ylabels(axs[:, 0])
fig.align_xlabels(axs[-1, :])
for ax in axs.flatten():
    ax.tick_params(axis='both', which='both', length=3.5)

# save_fig("scatter_matrix_plot")

plt.show()
```

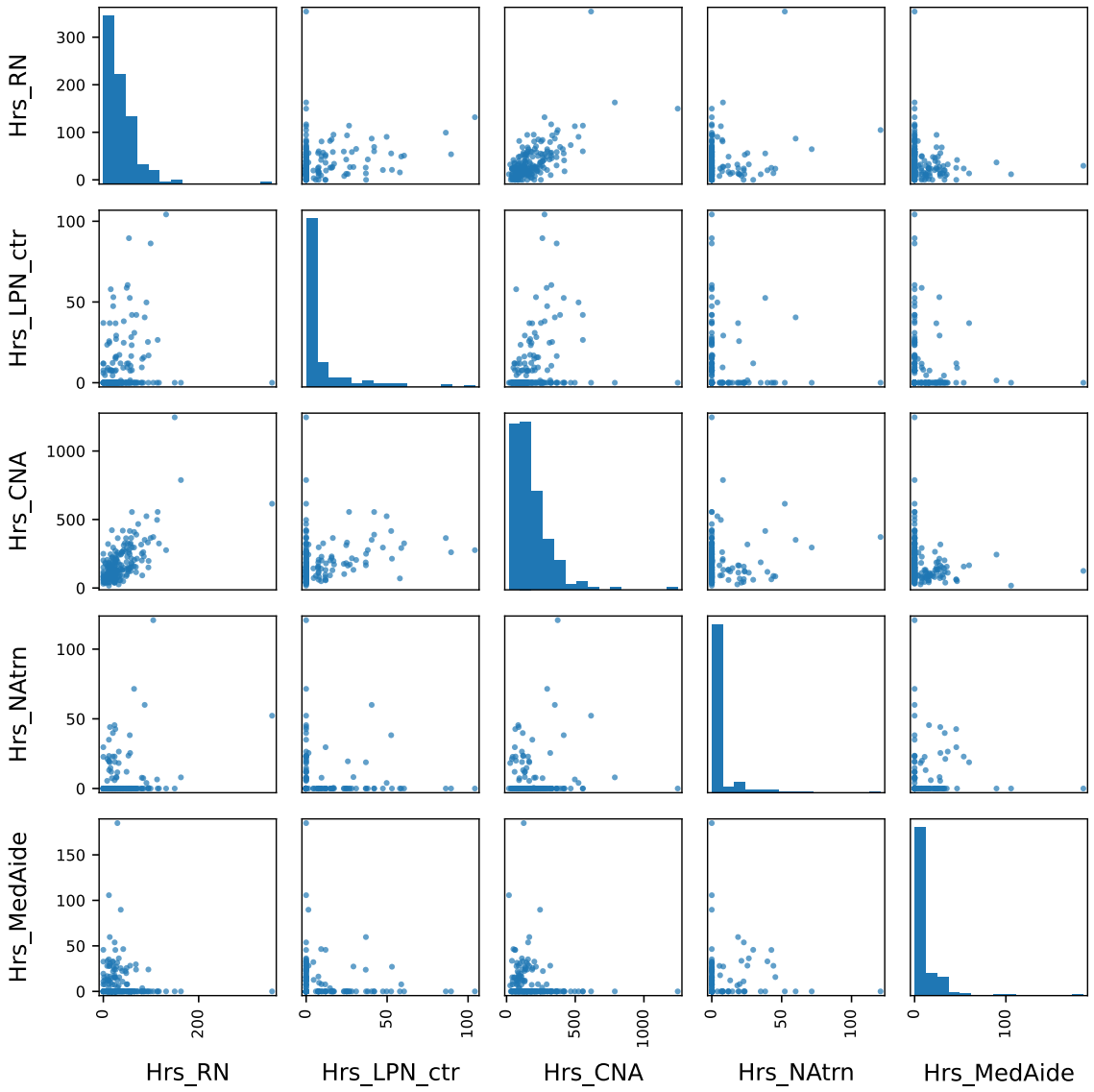

Figure 1: Scatter matrix of nursing worker working hours

Compare groups

> **i** Note 1: [Recommendation].

```
##| fig-subcap:
##|   - Average working hours with 95% confidence intervals.
##|   - Results of group comparisons by independent t-tests.
```

```
N_GROUPS = 6
N_LEVELS = 1

data_ = np.random.normal(loc=5, scale=3.0, size=(N_GROUPS, N_LEVELS, 10))

# Calculate averages and confidence intervals
averages = np.mean(data_, axis=2)
conf_intervals = np.zeros_like(averages, dtype=float)

for group_idx in range(N_GROUPS):
    for level_idx in range(N_LEVELS):
        interval = stats.t.interval(
            0.95,
            len(data_[group_idx, level_idx]) - 1,
            loc=np.mean(data_[group_idx, level_idx]),
            scale=stats.sem(data_[group_idx, level_idx])
        )

        # Use upper bound
        conf_intervals[group_idx, level_idx] = np.abs(
            interval[1] - averages[group_idx, level_idx]
        )

# -- Plot grouped bars with confidence intervals ------------------------------

width = 0.2
colors = plt.cm.Blues_r(np.linspace(.15, .85, N_LEVELS))
line_thickness = 0.6
stagger_amount = 0.8

fig, ax = plt.subplots()

for level_idx in range(N_LEVELS):
    bars = ax.bar(
        np.arange(N_GROUPS) + level_idx * width - (width * (N_LEVELS - 1) / 2),
        averages[:, level_idx],
        yerr=conf_intervals[:, level_idx],
        width=width,
        edgecolor="white",
        alpha=0.85,
        # capsize=3,
        color=colors[level_idx],
        error_kw={'elinewidth': line_thickness, 'capsize': 0},
        label=f'Level {level_idx + 1}',
    )

# Style
ax.set_ylabel('Values')
```

```
group_labels = [f'Group {i}' for i in range(1, N_GROUPS + 1)]
ax.set_xticks(np.arange(N_GROUPS))
ax.set_xticklabels(group_labels, rotation=60, ha='right')

# ax.legend(title='', bbox_to_anchor=(1.05, 1), loc='upper left')

# -- Add staggered sigbars and asterisks for select btwn-group comparisons ---

significance_level = 0.09
stagger_index = 0
stats_list = []

for comb in combinations(range(N_GROUPS), 2):
    group1_center = ax.get_xticks()[comb[0]]
    group2_center = ax.get_xticks()[comb[1]]

    t_stat, p_value = stats.ttest_ind(
        data_[comb[0], :, :].flatten(),
        data_[comb[1], :, :].flatten()
    )

    if p_value < significance_level:
        tallest_bar_height = np.max(averages) + np.max(conf_intervals) + 0.5

        # Adjust the stagger amount
        significance_height = (
            tallest_bar_height
            + np.max(conf_intervals) * 0.07
            + stagger_index * stagger_amount
        )

        # Plot staggered lines aligned with the midpoints of compared groups
        ax.plot(
            [group1_center, group2_center],
            [significance_height] * 2,
            color='black',
            lw=line_thickness
        )

        # Plot asterisks aligned with the center of the significance bars
        asterisks = (
            '*' * sum([p_value < alpha for alpha in [0.01, 0.001, 0.0001]])
        )
        ax.text(
            (group1_center + group2_center) / 2,
            significance_height,
            asterisks,
```

```
            ha='center',
            va='bottom',
            fontsize=10
        )

        # Increment the index for staggered bars
        stagger_index += 1

        # Store significant comparisons, t values, and sample sizes
        sample_size1 = len(data_[comb[0], :, :].flatten())
        sample_size2 = len(data_[comb[1], :, :].flatten())
        stats_list.append({
            "Comparison":
                f'{group_labels[comb[0]]} vs {group_labels[comb[1]]}',
            "p-value":
                f"{p_value:.4f}",
            "t-statistic":
                f"{t_stat:.4f}",
            "Sample Size": (
                f'{group_labels[comb[0]]} = {sample_size1}, '
                f'{group_labels[comb[1]]} = {sample_size2}'
            )
        })

# Style and show
ax.spines[['top', 'right']].set_visible(False)
ax.spines[['bottom', 'left']].set_visible(False)
ax.set_axisbelow(True)

ax.grid(axis='y')
plt.tight_layout()
plt.show()

stats_df = pd.DataFrame(stats_list)
```

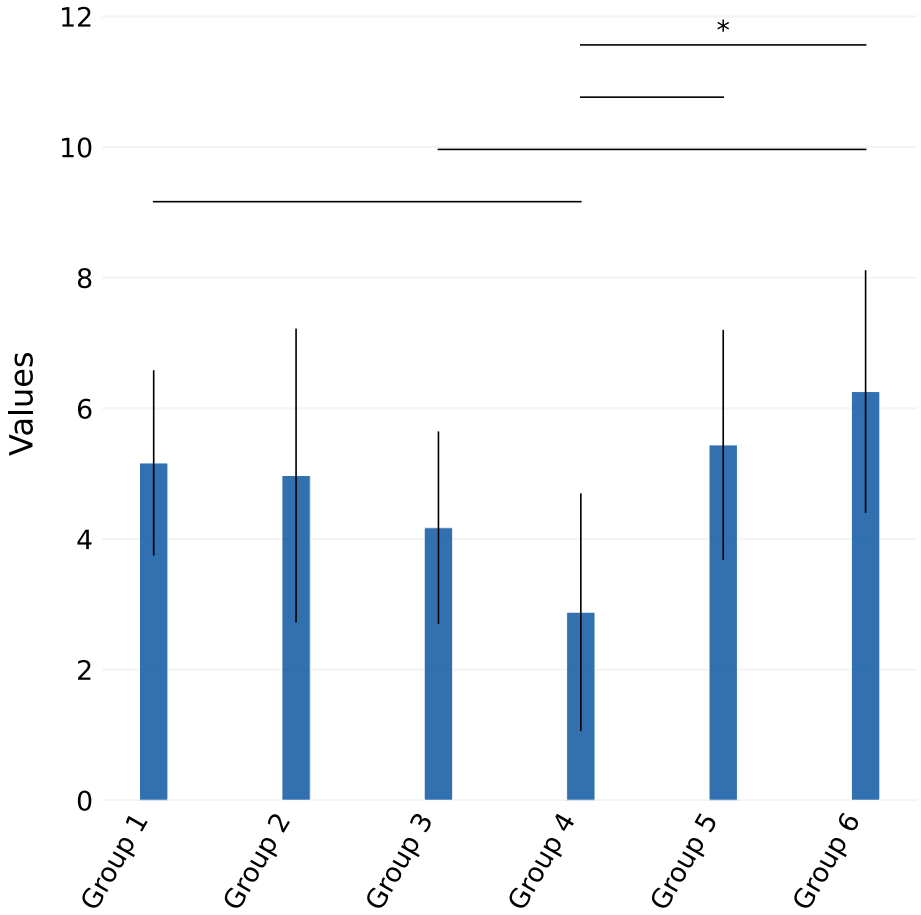

Figure 2: Comparison of average nurse working hours.

Comparison	p-value	t-statistic	Sample Size
Group 1 vs Group 4	0.0379	2.2406	Group 1 = 10, Group 4 = 10
Group 3 vs Group 6	0.0621	−1.9890	Group 3 = 10, Group 6 = 10
Group 4 vs Group 5	0.0344	−2.2892	Group 4 = 10, Group 5 = 10
Group 4 vs Group 6	0.0088	−2.9394	Group 4 = 10, Group 6 = 10

Table 2: Results of group comparisons by independent t-tests.

Feature engineer

Join geographical data

Join seasonal data

Analyze geography

Analyze seasonality

Model

Extra visualizations

Sparklines

```
# Plot sparklines of average work hours across 91 days by state
(
    GT(gt_df.head(), rowname_col="STATE")
    .fmt_nanoplot(
        columns="lines",
        reference_line="mean",
        reference_area=["min", "q1"]
    )
    .tab_header(
        title="Nurse hours worked in the United States",
        subtitle="The top 5 busiest states",
    )
    .tab_stubhead(label="State")
    .cols_label(
        lines="Total hours worked over 91 days",
    )
)
```

Nurse hours worked in the United States

The top 5 busiest states

State	Total hours worked over 91 days
2.80K2.87K2.84K2.46K	K2.38K2.91K2.91K2.98K2.97K2.89K2.47K2.37K2.88K2.88K2.89K2.90K2.80K2.36K2.24K
63.1K62.4K60.0K46.8K	K46.3K59.9K62.7K63.5K62.8K59.9K48.0K47.0K59.6K61.9K63.4K62.0K58.3K46.4K44.3K
53.5K52.4K50.8K40.0K	K39.0K50.4K52.6K52.7K52.0K49.6K39.6K38.7K50.3K53.1K54.0K53.5K51.5K39.9K37.7K
31.1K31.0K30.5K25.0K	K24.2K29.8K30.6K30.9K31.0K30.7K25.3K24.3K30.4K30.8K31.1K31.0K30.3K24.7K23.1K
K279K285K286K286K28	2K245K242K277K285K285K285K281K246K244K278K284K286K286K280K247K237K

Figure 3: Sparklines of average work hours across 91 days by state.

Concluding thoughts

(see Note 1)

Bibliography