Nurse Staffing Recommendations

Matthew Bain

2024-03-22

abstract I investigate nurse staffing data to provide informed recommendations to a medical staffing organization.

Table of contents

Imports	
The dataset	1
Some GT examples	
SQL	14
Bibliography	14

[...]

Imports

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

from great_tables import GT
from pandas.plotting import scatter_matrix

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

The dataset

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'
```

```
)
else:
print("data loaded.")
```

```
data.sample(5)
```

```
NAME
                 TY -TY - DateScenDON N ctrNA en A ctr NA-NA-NA-
               NAMEFIPS
                                                   tıtının entrum ctr
                              sus
     0751 PROTEET 3202 PO 402284 0.0 ... 0.00129.11629.160.0011.6311.63 0.0 0.00 0.00 0.0
                  ford
        LAGE
        GREEN
         OF
        BRIS-
         TOL
  134826
         RE-
        HAB
          &
       HEALTH
        CEN-
         TER
     14612BLY-LA IL Cook 312022Q2401232 8.0 ... 0.00183.593.590.00 0.00 0.00 0.0 0.00 0.00 0.0
  314429 MOUTRINGE
        PLACEARK
     TABANT Louis
        CREEK
       HEALTH
        AND
         RE-
  649459
         HA-
         BIL-
         ITA-
        TION
        CEN-
         TER
     395640HNMC-PA Al- 32022024012098 0.0 ... 100.2550.7224.5226.250.00 0.00 0.0 0.00 0.00 0.0
       JKANEK-
                legheny
         REESPORT
  992920 GIONAL
        CEN-
         TER-
         MC
     265793LA LA MO Ma-122024Q2402238 0.0 ... 0.0083.333.330.0011.751.75 0.0 14.3814.38 0.0
  661170 PLAPAATA
        NURS-
         ING
        HOME
```

```
# TODO: pivot on day

data_pivoted = data.pivot_table(
    index="STATE",
    columns="WorkDate",
    values="Hrs_RN",
    aggfunc='mean'
)

# Resetting the index for easier column access
# data_pivoted.reset_index(inplace=True)
data_pivoted
```

STATE

A\$5,66\$20**89**\$16**\$**4\$325**\$**\$20**5**6\$3**\$3**\$**7**\$6**000057**\$8**05**\$6**\$**933**\$**\$2.5**2**53**9\$**\$73**6**\$93**\$7**\$6**\$**93**\$7**\$6**\$**000005**7**\$6**\$**1\$6**\$**74000 AZ6.900050800048816034092796924364587#5013400869139038.312459490080043900802989500347495605579697085 AR .2.2445.04824703446532823504577.4834d1824387364608.3.382883289981363558709243202463149615753058571 A32.93873687202961858619823023166228884588545781949494.1.056339082947466278930848667884459934282662025288417 C29.643292332963833 6235338284282373246233251804753080.34.460476809473388182418384863547980805005720928120675 C37.22809860691019722572890964389685601936573757.11.452899857097856084728991812406019581608252113476 C35.5930187028485057878892068253203879099049859187882789.1895297876572780744151819918443512490569427478118387959 DT6.4 9239**59**21**79**552**92**441**73**576**8**5735**59**35**529**35**29**3 2647117.99.6 (9272**8**365**2.97**64**3**782**183**249**7**105**73**8300**93**82**50**165**8** 188235 D48.074948069773255580144494849953520979576176025827949.12517504882377370601464687485532871346512468382826744 F44.205075281235613511040H8674565561885365553123696.51.224894747245927853308758195674.656387487584860499813 G/8.5 2118**38221862213622136221494**20065226**86**32**2**28694606224.242886273**372**14968231**26**2406224986564**2**4729252027094 HD.10800c607078049000639026627882384040588424393.74658978578251984636878978978978928274563437317 I&2.838333962360505058689277932190452420563662217381.25.16458062494205094783285587254962449522389864409823 ID4.3236299839281892376247642587518122690960129475.227.762275729387248162230046400650232455023550263793000 ILO.5 188784528805464215158110228482015140055646539532447.1072782901564991602047878478467808993881744886754933 ID5.732614727346267346684251462626628023669432617402947.254576445384238926738232425246069005504230693640 K20.9927**5**235725075234948442597782218436521557322.88235922460267221854464277823596929217205638 KX0.268725329852486028436069\$449505636846\$332**9**925.3.25889**0**69505226\$4863**8**249521**63**21953163853472 L**A**.65**0920137939577229479341177582.2225566**077326051.0.84**215290**562**3038619866629G05239**4**38B**602**249793**9980 MID.48509085527573565659544848225680952163147975505.4.6952156183899857547528562597514027136123693727 M44.60566785329020965063448355556519505219855586469.48.3370366423492355995542685559658087284833999012 M1.840403695907633656320330803744872436659637.47.4357798572866530707882689076782850436735043649 MIST. 0.34130 2427 **8 1**815**68**1819**0 78**4550*6*8507**89**75592495527420487 8 9.42 . 7*6*063 **5 2**3557424121 24444**0 6**746**36**741 2**8**156**8** 7848 **2 2**595**6**5 9 1 7 M(5).7884943794386361815637651667438816487828128.7.21161847750051033664863618156377621602955683642 M29.8277307028764028349871372982901370207569028792701015502.8.73015592184978205023394019552195499351007979444968100 MES.27344057445200065384928846819885772974458519591533.90843783825680822076576222259844752742945929831 N2.2479966346559290496902223125154265962685292627.26056622392364867922648679205852068521027049 ND .8692746024333023524658280493502423564209278445.9.7.8558349158699788853467562871399885649256993243

<u>tarrocaro altra e de la composição de la colocaro de la colocaron de la colocaron</u>

data_pivoted.iloc[:, 1:]

STATE

<u> 1900: Activación de la lacación de la lacación de la company de la com</u>

```
# (
#
     GT(data pivoted, rowname col="STATE")
#
     .fmt nanoplot(
        columns=data_pivoted.columns[1:],
#
#
         reference line="mean",
#
         reference area=["min", "q1"]
#
    )
    .fmt_nanoplot(
#
       columns=data pivoted.columns[1:],
#
#
         plot type="bar",
#
         reference line="max",
        reference_area=["max", "median"]
#
    )
# )
```

```
data.describe().round(1)
# display(Markdown(data.describe().to_markdown()))
```

COUNWork-MF3-RNs-RNs-RNs-RRs-RRs-RN HzsIzs-PCN-Ars_ICrs_CHrs-Hrs-NAddeAnde_ctr
TY_Dat&cen00000000_ctrNad-Nad-NadFIPS sus minin_emmips_ctr tum_entups_ctr

```
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()
```

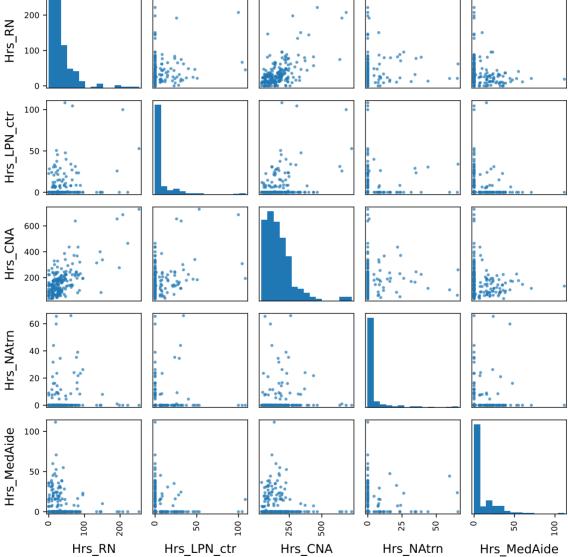


Figure 1: Scatter matrix of nursing worker working hours

```
import pandas as pd
df = pd.DataFrame({'name': ['arizona', '', 'berlin', 'london']})
```

```
from geopy.geocoders import Nominatim
geolocator = Nominatim(user_agent="geo_clipboard")

from geopy.extra.rate_limiter import RateLimiter
geocode = RateLimiter(geolocator.geocode, min_delay_seconds=1)
df['location'] = df['name'].apply(geocode)

df['point'] = df['location'].apply(lambda loc: tuple(loc.point) if loc else None)
```

df

	name	location	point
0	arizona	(Arizona, United States, (34.395342, —111.7632 (34.395342,	4.395342, -111.763275, 0.0)
1		None	None
2	berlin	(Berlin, Deutschland, (52.510885, (5 13.3989367))	2.510885, 13.3989367, 0.0)
3	london	(London, Greater London, England, (51.4893) United Kingd	3335, -0.14405508452768728, 0.0)

```
from great_tables import GT

df = data.loc[150000:, [
    "STATE",
    "COUNTY_NAME", "COUNTY_FIPS",
    "CITY",
    "PROVNAME", "PROVNUM",
    # "MDScensus"

]].value_counts().reset_index()
GT(df.head(n=10))
```

STATE	COUNTY NAME	COUNTY FIPS	CITY	PROV- NAME	PROVNUM	count
CA	Alameda	1	ALAMEDA	ALAMEDA HEALTH- CARE & WELL- NESS CEN- TER	555486	91
ОН	Mahoning	99	AUSTIN- TOWN	AVEN- TURA AT HUMILITY HOUSE	366186	91
ОН	Lucas	95	TOLEDO	OHIO LIV- ING SWAN CREEK	365996	91
ОН	Lucas	95	TOLEDO	OTTER- BEIN SUNSET HOUSE	366148	91
ОН	Lucas	95	TOLEDO	PARK TER- RACE NURSING AND RE- HABILITA- TION CEN- TER	365339	91
ОН	Lucas	95	TOLEDO	POINT PLACE HEALTH- CARE AND REHABIL- ITATION CENTER	366039	91
ОН	Lucas Mudis on	95 95	WATER- VILLE WAITER- LONDON MOUSE	ASTORIA PLACE OF AYDEN MENTER CARE OF CORRESE CORRESE	365747 36584	91 —— 91 —
			11	VILLE VILLE		

Some GT examples

```
from typing import Any
from IPython.display import display as ipy_display, HTML
import numpy as np
def display2(
    *args,
    globs: dict[str, Any] | None = None,
    bold: bool = True,
   width: str = "400px" # Fixed width for each block
) -> None:
    11 11 11
   Display an informative representation of multiple objects side-by-side in
Jupyter.
    Parameters
    *args : tuple
       Tuple of expressions to evaluate and display.
    globs : dict[str, Any], default=None
        Global namespace, to give eval() access to nonlocals passed by name.
    bold : bool, default=True
       Option to enable/disable string styling.
    width : str, default="400px"
       Fixed width for each displayed block in the Jupyter notebook.
    Warnings
    This function uses `eval()` to render expressions it receives
    as strings. Access to variables in the global namespace is controlled
    by `globs`. Take care to only pass trusted expressions to the function.
    11 11 11
    if globs is None:
        globs = \{\}
    outputs = []
    for arg in args:
        name = f"<b>{arg}</b>" if bold else arg
        value = np.round(eval(arg, globs), 2)
        shape = np.shape(value)
                   content = f"<div style='width:{width}; padding:10px;</pre>
float:left;'>{name}\n--- {repr(shape)} ---\n{repr(value)}</div>"
        outputs.append(content)
    # Clearfix for layout
    clearfix = "<div style='clear: both;'></div>"
```

```
# Display the HTML content in Jupyter
html_output = ''.join(outputs) + clearfix
ipy_display(HTML(html_output))
return None
```

```
A = np.array([[1, 3], [2, 4]])
x = np.array([[0, 1]])

display2(
    "A", "x.T", "np.dot(A, x.T)", globs=globals(), bold=True, width="100px"
)
```

<IPython.core.display.HTML object>

```
display2(
    "data['STATE'].value_counts()",
    "data['COUNTY_NAME'].value_counts()",
    "data['CITY'].value_counts()",
    "data['PROVNAME'].value_counts()",
    "data['MDScensus'].value_counts()",
    width="340px",
    globs=globals()
)
```

```
<IPython.core.display.HTML object>
```

```
data[["CY_Qtr", "WorkDate", "MDScensus"]]
```

	CY_Qtr	WorkDate	MDScensus
0	2024Q1	20240101	50
1	2024Q1	20240102	49
2	2024Q1	20240103	49
3	2024Q1	20240104	50
4	2024Q1	20240105	51
•••	•••	•••	•••
1330961	2024Q1	20240327	81
1330962	2024Q1	20240328	83
1330963	2024Q1	20240329	85
1330964	2024Q1	20240330	82
1330965	2024Q1	20240331	82

SQL Bibliography