Final Project

Esteban Mendoza

March 2020

1 Abstract

Contents

1 Abstract **1**

2 Introduction **3**

3 Materials & Methods **3**
3.1 Script 3
3.2 Raw Data Visuals 7
3.2.1 Code 7
3.2.2 Graphical Data 11

4 Results **12**

5 Discussion **12**

2 Introduction

A disease is a condition that negatively affects the structure and hinders the homeostatic functions of an organism. Disease is characterized by specific symptoms exhibited by the affected organism. As medical technology has grown, so has the access to treatments that relieve the symptoms and repair the malfunctions caused by disease. Unfortunately, the costs to develop and employ these innovations have proven to be expensive and unaffordable for most people seeking treatment. In a society where resources to expend on healthcare are scarce, it is important to question how we can effectively allocate limited resources to maintain the health of the general public. Diseases can be categorized into different classifications, such as communicable and non-communicable. Additionally, these diseases can be categorized as infectious, deficiency, hereditary, and physiological diseases. However, assigning "disease" to a condition is a subjective topic. Studies have shown that different factors account for whether people believe themselves to be ill. Some of these factors include class, gender, ethnic group "and less obvious factors such as proximity to support from family members" [**?**]. Additionally, as expectations of health change throughout time, so does the classification of something as a disease.

For example, osteoporosis was "officially recognized as a disease by the WHO in 1994" [**?**]. This classification changed osteoporosis from a "normal part of aging" to a recognized pathological condition. Homosexuality has also had a history in the classification of disease. In the early 20th century, homosexuality was considered an endocrine disorder, then later classified as a mental disorder, and then finally "de-pathologized" in 1974. It's important to properly identify a condition as a disease in order to properly allocate resources for treatment, while also being conscious of the weight and stigmatization that the label "disease" might carry.

In the analysis of this dataset, I attempt to identify what different members of society classify as a disease, and how much public funding should go into their management. The data used to inform this analysis was collected in a survey form. The survey was sent out to Finnish laypeople, doctors, nurses, and parliament members. The purpose of the survey was to collect opinions on different states of being and identify how these people classified them.

3 Materials & Methods

3.1 Script

The following script works from the shell. It asks the user to select a "state of being." Then the user is asked to select a rank from 1-5. The rank they choose will produce a graph that highlights the percentage of people (doctors, nurses, laypeople, and parliament members) that considered the selected "state of being" as a disease (1-2: not a disease; 3: neutral; 4-5: yes a disease).

```
#!/usr/bin/env python
```

```
# coding: utf-8

# In[ ]:


#!//home/eebc177student/anaconda3/bin/python3
#written script will work anywhere in shell

import numpy as np
import pandas
import os
datapath = "/home/eebc177student/Developer/repos/eeb-c177
    -project/analyses"
directory = '/home/eebc177student/Developer/repos/eeb-
    c177-project/analyses'
os.chdir(directory)
#set working directory in analyses directory bc that's
    where the csv file is


import csv
import re
data = pandas.read_csv('final_data.csv')
data = data.rename(columns={'Restless Legs SyndromeA.1':'
    Restless Legs SyndromeB', 'Personality DisorderA.1':'
    Personality DisorderB'}) #some columns were oddly
    names
columns = list(data.columns) #turn the columns into a
    list
columns = columns[4:-1] #only use columns from these
    indexes bc they are the conditions

#A = is X a disease (rank from 1-5)
regex = re.compile(r'[\w\s]*[^A]A{1}$')
columnsA = list(filter(regex.match,columns))

#B = should public funding be used for X (rank 1-5)
regex = re.compile(r'[\w\s]*[^B]B{1}$')
columnsB = list(filter(regex.match,columns))


dis = str(input("what disease do you want to compare?: ")
    ) #ask user what disease to evaluate
dis = dis.upper() #make input uppercase bc column names
    uppercase
dis = dis.split()
```

```
error = False #for/if statement corrects for incorrect
    name input
for word in dis:
    if word in columnsA:
        pass
    else:
        error = True
if error: print('Please enter a valid condition')
else:
    def listtostring(s): #
        str1 = " "
        return(str1.join(s))
    dis = listtostring(dis)
    df = pandas.DataFrame(data)
    df = df[['Group', dis]] #extract data columns for
        people surveyed, and disease selected by user

    from collections import Counter

    classify = str(input('Percent of participants that
        considered {} as a disease on a scale from 1-5: '.
        format(dis)))
    #find percent of people who classified X as a disease
         on a scale from 1-5
    #choose a number on the scale
    error = False #for/if statement corrects for
        incorrect rank input
    number = [str(i) for i in range(1,6)]
    for rank in classify:
        if rank in number:
            pass
        else:
            error = True
    if error: print('please enter a rank from 1-5')
    else:
        classify = float(classify)


        def profession(data, person):
            person_HArank = df.values.tolist() #make the
                people and classifications into a list
            HA5_people = person_HArank.count([person,
                classify]) #count list items that include
                groups of people that ranked X condition
                as X rank
```

```
    person_HArank = [tuple(i) for i in
        person_HArank] #tuple instead of list
        itmes
    counts = Counter(x[0] for x in person_HArank)
         #count number of tuple items
    total_people = counts[person] #count all
        people in the survey
    percentage = HA5_people/total_people*100 #
        calculate percentage of people who ranked
        X condition as X rank
    return percentage

#different groups of people from data
layperson = profession(data, 'Layperson')
nurse = profession(data, 'Nurse')
doctor = profession(data, 'Doctor')
parliament = profession(data, 'Parliament')

import matplotlib.pyplot as plt
#%matplotlib inline
plt.style.use('ggplot')
#make a bar graph
def plot_percentage_person(layperson, doctor,
    nurse, parliament):
    #function to plot people and percentages
    x = ['Layperson', 'Doctor','Nurse','
        Parliament'] #people on the x axis
    percent = [layperson, doctor, nurse,
        parliament] #percentages to be calculated
        per person
    x_pos = [i for i, _ in enumerate(x)] #add
        groups of people
#bar graph settings
    plt.bar(x_pos, percent, color='green')
    plt.xlabel("Person")
    plt.ylabel("Percent")
    plt.title("Percent of professionals surveyed
        who classify {} \n as a rank {} disease on
        a scale of 1-5".format(dis, classify))



    plt.xticks(x_pos, x)


#plot the graph
```

```
        plt.show()
        return
    plot_percentage_person(layperson, doctor, nurse,
        parliament)
    #use the function
```

3.2 Raw Data Visuals

The following code was used to provide a visualization of averages from the raw data file. The figures produced show demographic rankings for the two survey questions asked. Figure 1 demonstrates the mean ranks for each of the "states of being" that each demographic would consider a disease. Ranks 4-5 indicate a disease, and ranks 1-2 indicate not a disease. Figure 2 demonstrates the mean ranks for each of the "states of being" that each demographic would consider using public funds for its management. Ranks 4-5 indicate support for the use of public funding, and ranks 1-2 indicate no public funding should be invested.

3.2.1 Code

```
library(ggplot2) #load library
library(dplyr) #load library
names(final_data) <- tolower(names(final_data)) #
    lowercase column titles
final_data <- as.data.frame(final_data) #make dataframe
names(final_data) <- gsub(x = names(final_data), pattern
    = "a\\.1", replacement = "b") #rename badly named
    column titles
names(final_data) <- gsub(x = names(final_data), pattern
    = "\\.", replacement = " ") #remove period in column
    titles
final_data$group <- tolower(final_data$group) #make
    values in 'group' column lowercase
rank <- readline(prompt = 'Ranks for disease or funding?:
    ') #ask user if they want a chart for disease rank or
     funding rank
rank <- tolower(rank) #make answer lowercase

if (rank == 'disease') { #if user selected 'disease':
  parliament <- filter(final_data, group == tolower('
      parliament')) #take data matching demographic '
      parliament'
  parliament <- select(parliament, ends_with('a')) #
      selecting columns 'states of being' that are disease
       rank responses since they end in 'a'
```

```
names(parliament) <- gsub(x = names(parliament),
    pattern = "a$", replacement = "") #remove the 'a'
doctor <- filter(final_data, group == tolower('doctor')
    ) #take data matching demographic 'doctor'
doctor <- select(doctor, ends_with('a'))
names(doctor) <- gsub(x = names(doctor), pattern = "a$"
    , replacement = "")
nurse <- filter(final_data, group == tolower('nurse'))
    #take data matching demographic 'nurse'
nurse <- select(nurse, ends_with('a'))
names(nurse) <- gsub(x = names(nurse), pattern = "a$",
    replacement = "")
layperson <- filter(final_data, group == tolower('
    layperson')) #take data matching demographic '
    layperson'
layperson <- select(layperson, ends_with('a'))
names(layperson) <- gsub(x = names(layperson), pattern
    = "a$", replacement = "")


par_mean <- colMeans(parliament) #take means for each '
    state of being' ranked by this demographic
doc_mean <- colMeans(doctor)
nur_mean <- colMeans(nurse)
lay_mean <- colMeans(layperson)

lay_mean <- as.data.frame(lay_mean) #make layperson
    means and states of being into a datafram

lay_mean$nurse <- nur_mean #add nurse means to
    dataframe
lay_mean$doctor <- doc_mean #add doctor means to
    dataframe
lay_mean$parliament <- par_mean #add parliament means
    to dataframe

means <- as.data.frame(lay_mean) #rename dataframe as
    means

df <- tibble::rownames_to_column(means, "VALUE") #'
    states of being' to column

library(reshape2) #load library
df <- melt(df, id.vars="VALUE") #stack a set of columns
     into a single column of data
```

```
  ggplot(data=df, aes(x=VALUE,y=value,fill=variable)) + #
      states of being on x axis, ranks on x axis, group
      means as bars
    geom_bar(position="dodge",stat="identity") +
    coord_flip() + labs(y= "Mean Rank (1-5)", x = "States
        of Being") + scale_fill_discrete(name = "
        Demographic", labels = c("Layperson", "Nurse", "
        Doctor", "Parliament")) + #flip graph so bar plots
         run horizontal and add titles
    theme(plot.title = element_text(hjust = 0.5)) +
    ggtitle("Mean Demographic Ranks: Ranking 'States of
        Being' as Diseases")
} else if (rank == 'funding') { #do this if user asked
    for funding
  parliament <- filter(final_data, group == tolower('
      parliament')) #take data matching demographic '
      parliament'
  parliament <- select(parliament, ends_with('b')) #
      selecting columns 'states of being' that are funding
       rank responses since they end in 'b'
  names(parliament) <- gsub(x = names(parliament),
      pattern = "b$", replacement = "") #remove the 'b'
  doctor <- filter(final_data, group == tolower('doctor')
      )
  doctor <- select(doctor, ends_with('b'))
  names(doctor) <- gsub(x = names(doctor), pattern = "b$"
      , replacement = "")
  nurse <- filter(final_data, group == tolower('nurse'))
  nurse <- select(nurse, ends_with('b'))
  names(nurse) <- gsub(x = names(nurse), pattern = "b$",
      replacement = "")
  layperson <- filter(final_data, group == tolower('
      layperson'))
  layperson <- select(layperson, ends_with('b'))
  names(layperson) <- gsub(x = names(layperson), pattern
      = "b$", replacement = "")


  par_mean <- colMeans(parliament)
  doc_mean <- colMeans(doctor)
  nur_mean <- colMeans(nurse)
  lay_mean <- colMeans(layperson)

  lay_mean <- as.data.frame(lay_mean)

  lay_mean$nurse <- nur_mean
```

```
  lay_mean$doctor <- doc_mean
  lay_mean$parliament <- par_mean

  means <- as.data.frame(lay_mean)

  df <- tibble::rownames_to_column(means, "VALUE")

  library(reshape2)
  df <- melt(df, id.vars="VALUE")

  ggplot(data=df, aes(x=VALUE,y=value,fill=variable)) +
    geom_bar(position="dodge",stat="identity") +
    coord_flip() + labs(y= "Mean Rank (1-5)", x = "States
        of Being") + scale_fill_discrete(name = "
        Demographic", labels = c("Layperson", "Nurse", "
        Doctor", "Parliament")) +
    theme(plot.title = element_text(hjust = 0.5)) +
    ggtitle("Mean Demographic Ranks: Ranking 'States of
        Being' as Warranting Public Funding")
} else { #if 'funding' or 'disease' not inputted
  print("please enter 'disease' or 'funding'") #print
      error message
}
```

3.2.2 Graphical Data

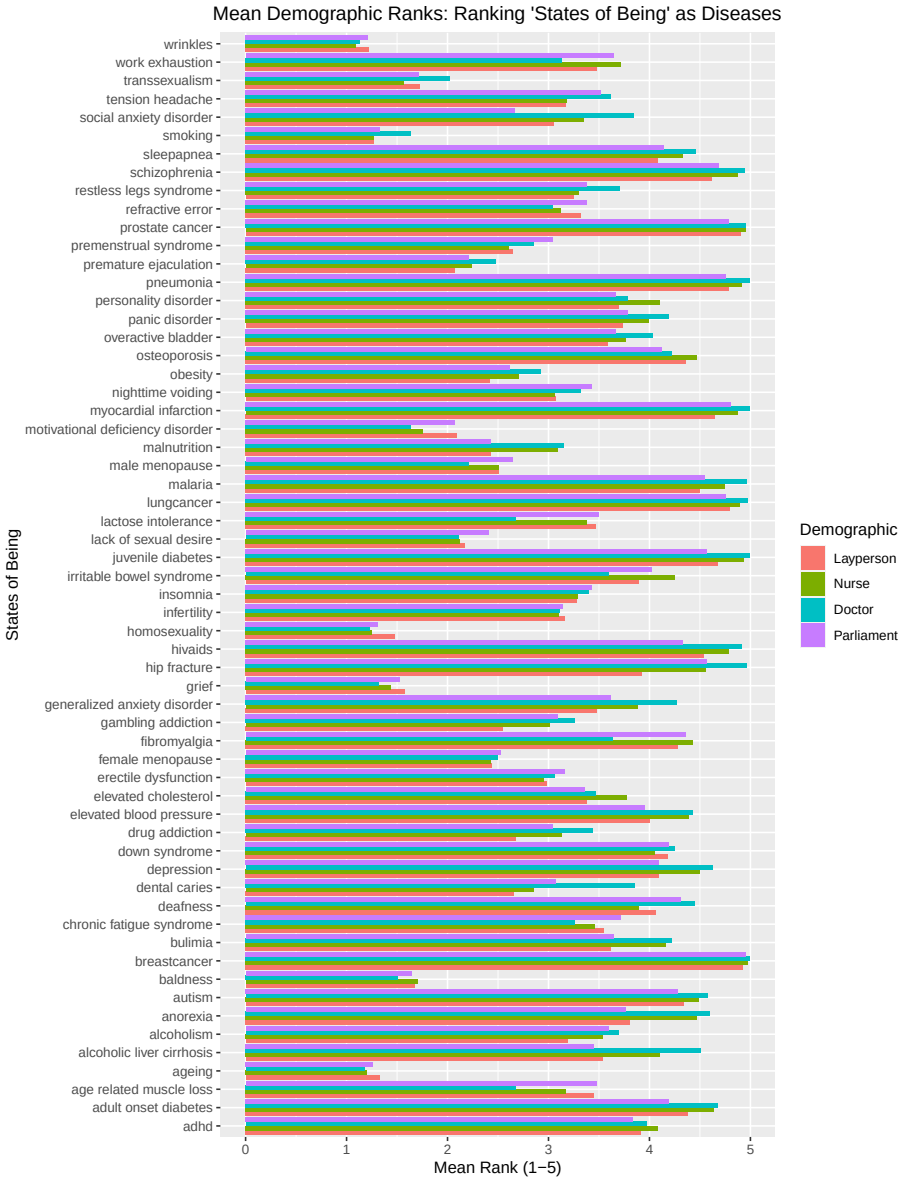

Figure 1: Demographic rankings for disease consideration

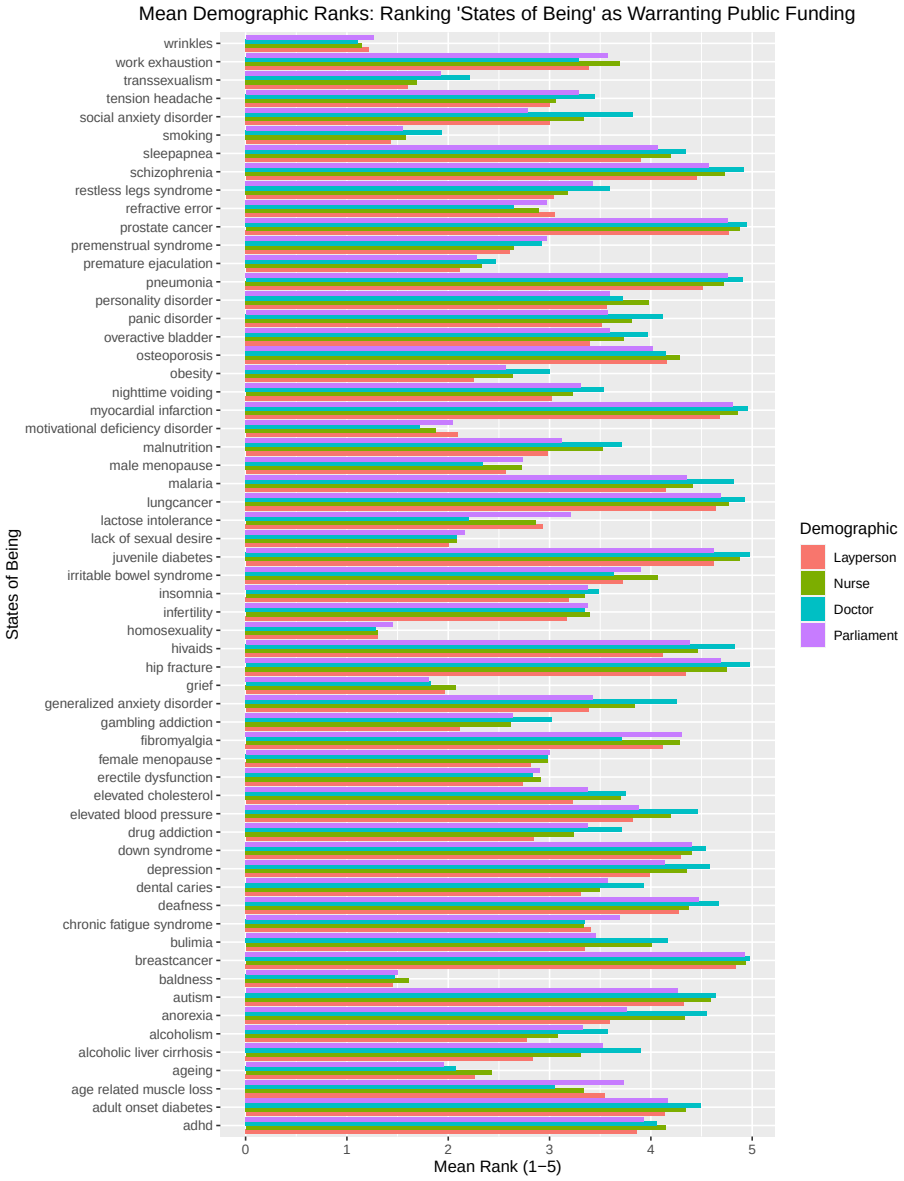

Figure 2: Demographic rankings for funding consideration

4 Results

5 Discussion