**WAFFLE CHART**

import numpy as np

import pandas as pd

from PIL import Image # converting images into arrays

import matplotlib as mpl

import matplotlib.pyplot as plt

import matplotlib.patches as mpatches # needed for waffle Charts

mpl.style.use('ggplot')

df = pd.DataFrame({

'Name\_of\_State': ['Andaman and Nicobar Islands','Andhra Pradesh','Arunachal Pradesh','Assam',

'Bihar','Chandigarh','Chhattisgarh','Delhi','Goa','Gujarat','Haryana','Himachal Pradesh' ,

'Jammu and Kashmir','Jharkhand','Karnataka','Kerala','Ladakh','Madhya Pradesh','Maharashtra'

,'Manipur','Meghalaya','Mizoram','Odisha','Puducherry','Punjab','Rajasthan','Tamil Nadu','Tripura'

,'Uttarakhand','Uttar Pradesh','Telengana','West Bengal'],

'Total\_Confirmed\_cases': [33,1650,1,43,528,102,58,4898,7,5804,517,41,726,115,651,500,41,2942,14541,2,12,1,169,8

,1233,3061,3550,29,60,2766,1085,1259]

})

def create\_waffle\_chart(categories, values, height, width, colormap, value\_sign=''):

# compute the proportion of each category with respect to the total

total\_values = sum(values)

category\_proportions = [(float(value) / total\_values) for value in values]

# compute the total number of tiles

total\_num\_tiles = width \* height # total number of tiles

print ('Total number of tiles is', total\_num\_tiles)

# compute the number of tiles for each catagory

tiles\_per\_category = [round(proportion \* total\_num\_tiles) for proportion in category\_proportions]

# initialize the waffle chart as an empty matrix

waffle\_chart = np.zeros((height, width))

# define indices to loop through waffle chart

category\_index = 0

tile\_index = 0

# populate the waffle chart

for col in range(width):

for row in range(height):

tile\_index += 1

# if the number of tiles populated for the current category

# is equal to its corresponding allocated tiles...

if tile\_index > sum(tiles\_per\_category[0:category\_index]):

# ...proceed to the next category

category\_index += 1

# set the class value to an integer, which increases with class

waffle\_chart[row, col] = category\_index

# instantiate a new figure object

fig = plt.figure()

# use matshow to display the waffle chart

colormap = plt.cm.coolwarm

plt.matshow(waffle\_chart, cmap=colormap)

plt.colorbar()

# get the axis

ax = plt.gca()

# set minor ticks

ax.set\_xticks(np.arange(-.5, (width), 1), minor=True)

ax.set\_yticks(np.arange(-.5, (height), 1), minor=True)

# add dridlines based on minor ticks

ax.grid(which='minor', color='w', linestyle='-', linewidth=2)

plt.xticks([])

plt.yticks([])

# compute cumulative sum of individual categories to match color schemes between chart and legend

values\_cumsum = np.cumsum(values)

total\_values = values\_cumsum[len(values\_cumsum) - 1]

# create legend

legend\_handles = []

for i, category in enumerate(categories):

if value\_sign == '%':

label\_str = category + ' (' + str(values[i]) + value\_sign + ')'

else:

label\_str = category + ' (' + value\_sign + str(values[i]) + ')'

color\_val = colormap(float(values\_cumsum[i])/total\_values)

legend\_handles.append(mpatches.Patch(color=color\_val, label=label\_str))

# add legend to chart

plt.legend(

handles=legend\_handles,

loc='lower center',

ncol=len(categories),

bbox\_to\_anchor=(0., -0.2, 0.95, .1)

)

width = 40

height = 10

categories = df.Name\_of\_State # categories

values = df['Total\_Confirmed\_cases'] # correponding values of categories

colormap = plt.cm.coolwarm # color map class

create\_waffle\_chart(categories, values, height, width, colormap) # call the function

Total number of tiles is 400

<Figure size 432x288 with 0 Axes>

