**TECHNOLOGY NAME: *Data Analysis with Cognos***

**PROJECT: *Public Health Awareness Campaign Analysis***

**PHASE NUMBER:4**

**INTRODUCTION:**

***In this phase we will continue building our project.***

* ***Continue building the analysis by creating visualizations using IBM Cognos and integrating code for data analysis.***
* ***Design dashboards and reports in IBM Cognos to visualize campaign reach, awareness levels, and impact metrics.***
* ***Use code Python to perform advanced data analysis, such as calculating engagement rates, conducting demographic analysis, or running statistical tests.***

**CODE:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import os

os.listdir("/kaggle/input/")

df = pd.read\_csv("/kaggle/input/mental-health-in-tech-survey/survey.csv")

df.shape

df.info()

df.isna().sum()

self\_employed\_percent = (df["self\_employed"].isnull().sum()/len(df["self\_employed"]))\*100

work\_interfere\_percent = (df["work\_interfere"].isnull().sum()/len(df["work\_interfere"]))\*100

print(f"The percentage of missing values in self\_employed column is **{**round(self\_employed\_percent, 2)**}**%")

print(f"The percentage of missing values in work\_interfere column is **{**round(work\_interfere\_percent, 2)**}**%")

df["self\_employed"] = df["self\_employed"].fillna(df["self\_employed"].mode()[0])

df["work\_interfere"] = df["work\_interfere"].fillna(df["work\_interfere"].mode()[0])

df.drop(["state", "comments"], axis=1, inplace=True)

df.isna().sum()

df.columns

plt.figure(figsize=(17,5))

ax = sns.countplot(x='Country', data=df)

ax.set\_xticklabels(

ax.get\_xticklabels(),

rotation=45,

horizontalalignment='right'

for p **in** ax.patches:

ax.annotate(p.get\_height(), (p.get\_x()+0.25, p.get\_height()+0.01), ha='center')

min\_age = df["Age"].min()

max\_age = df["Age"].max()

mean\_age = df["Age"].mean()

median\_age = df["Age"].median()

print(f"Min: **{**min\_age**}**, **\n**Max: **{**max\_age**}**, **\n**Mean: **{**mean\_age**}**, **\n**Median: **{**median\_age**}**")

df["Age"].unique()

negative\_age = (df["Age"]<0).sum()

over\_age = (df["Age"]>80).sum()

print(f"Number of negative age entries: **{**negative\_age**}\n**Number of overage: **{**over\_age**}**")

df.loc[df.Age<0, ["Age"]] = df["Age"].median()

df.loc[df.Age>80, ["Age"]] = df["Age"].median()

df["Age"].unique()

df.loc[df.Age<18, ["Age"]] = df["Age"].median()

df["Age"].hist()

import statistics

variance\_age = df["Age"].var()

standard\_dev\_age = statistics.stdev(df["Age"])

print(f"Mean: **{**round(mean\_age, 2)**}**"

f"**\n**Variance: **{**round(variance\_age, 2)**}**"

f"**\n**Standard Deviation: **{**round(standard\_dev\_age, 2)**}**")

sns.set\_style("whitegrid")

plt.figure(figsize = (8,5))

plt.title('Get Treatment of Survey Respondents', fontsize=12, fontweight='bold')

*#eda\_percentage = df['treatment'].value\_counts(normalize = True).rename\_axis('treatment').reset\_index(name = 'Percentage')*

ax = sns.barplot(x = 'treatment', y = 'Percentage', data = s4.head(10), palette='Purples')

for p **in** ax.patches:

width = p.get\_width()

height = p.get\_height()

x, y = p.get\_xy()

ax.annotate(f'**{**height**:**.0%**}**', (x + width/2, y + height\*1.02), ha='center', fontweight='bold')

def show\_shap(col, shap\_values=shap\_values, label=main\_label, X\_test=X\_test, ylabel='SHAP value'):

df\_infl = X\_test.copy()

df\_infl['shap\_'] = shap\_values[:,df\_infl.columns.tolist().index(col)]

gain = round(df\_infl.groupby(col).mean()['shap\_'],4)

gain\_std = round(df\_infl.groupby(col).std()['shap\_'],4)

cnt = df\_infl.groupby(col).count()['shap\_']

dd\_dict = {'col': list(gain.index), 'gain': list(gain.values), 'gain\_std': list(gain\_std.values), 'count': cnt}

df\_res = pd.DataFrame.from\_dict(dd\_dict).sort\_values('gain', ascending=False).set\_index('col')

plt.figure(figsize=(9,6))

plt.errorbar(df\_res.index, df\_res['gain'], yerr=df\_res['gain\_std'], fmt="o", color="r")

plt.title(f'SHAP values for column **{**col**}**, label **{**label**}**')

plt.ylabel(ylabel)

plt.tick\_params(axis="x", rotation=90)

plt.show();

print(df\_res)

return

for col **in** X\_test.columns:

print()

print(col)

print()

show\_shap(col, shap\_values, label=main\_label, X\_test=X\_test)

output:

['mental-health-in-tech-survey']

(1259, 27)

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 1259 entries, 0 to 1258

Data columns (total 27 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Timestamp 1259 non-null object

1 Age 1259 non-null int64

2 Gender 1259 non-null object

3 Country 1259 non-null object

4 state 744 non-null object

5 self\_employed 1241 non-null object

6 family\_history 1259 non-null object

7 treatment 1259 non-null object

8 work\_interfere 995 non-null object

9 no\_employees 1259 non-null object

10 remote\_work 1259 non-null object

11 tech\_company 1259 non-null object

12 benefits 1259 non-null object

13 care\_options 1259 non-null object

14 wellness\_program 1259 non-null object

15 seek\_help 1259 non-null object

16 anonymity 1259 non-null object

17 leave 1259 non-null object

18 mental\_health\_consequence 1259 non-null object

19 phys\_health\_consequence 1259 non-null object

20 coworkers 1259 non-null object

21 supervisor 1259 non-null object

22 mental\_health\_interview 1259 non-null object

23 phys\_health\_interview 1259 non-null object

24 mental\_vs\_physical 1259 non-null object

25 obs\_consequence 1259 non-null object

26 comments 164 non-null object

Timestamp 0

Age 0

Gender 0

Country 0

state 515

self\_employed 18

family\_history 0

treatment 0

work\_interfere 264

no\_employees 0

remote\_work 0

tech\_company 0

benefits 0

care\_options 0

wellness\_program 0

seek\_help 0

anonymity 0

leave 0

mental\_health\_consequence 0

phys\_health\_consequence 0

coworkers 0

supervisor 0

mental\_health\_interview 0

phys\_health\_interview 0

mental\_vs\_physical 0

obs\_consequence 0

The percentage of missing values in self\_employed column is 1.43%

The percentage of missing values in work\_interfere column is 20.97%

Timestamp 0

Age 0

Gender 0

Country 0

self\_employed 0

family\_history 0

treatment 0

work\_interfere 0

no\_employees 0

remote\_work 0

tech\_company 0

benefits 0

care\_options 0

wellness\_program 0

seek\_help 0

anonymity 0

leave 0

mental\_health\_consequence 0

phys\_health\_consequence 0

coworkers 0

supervisor 0

mental\_health\_interview 0

phys\_health\_interview 0

mental\_vs\_physical 0

Index(['Timestamp', 'Age', 'Gender', 'Country', 'self\_employed',

'family\_history', 'treatment', 'work\_interfere', 'no\_employees',

'remote\_work', 'tech\_company', 'benefits', 'care\_options',

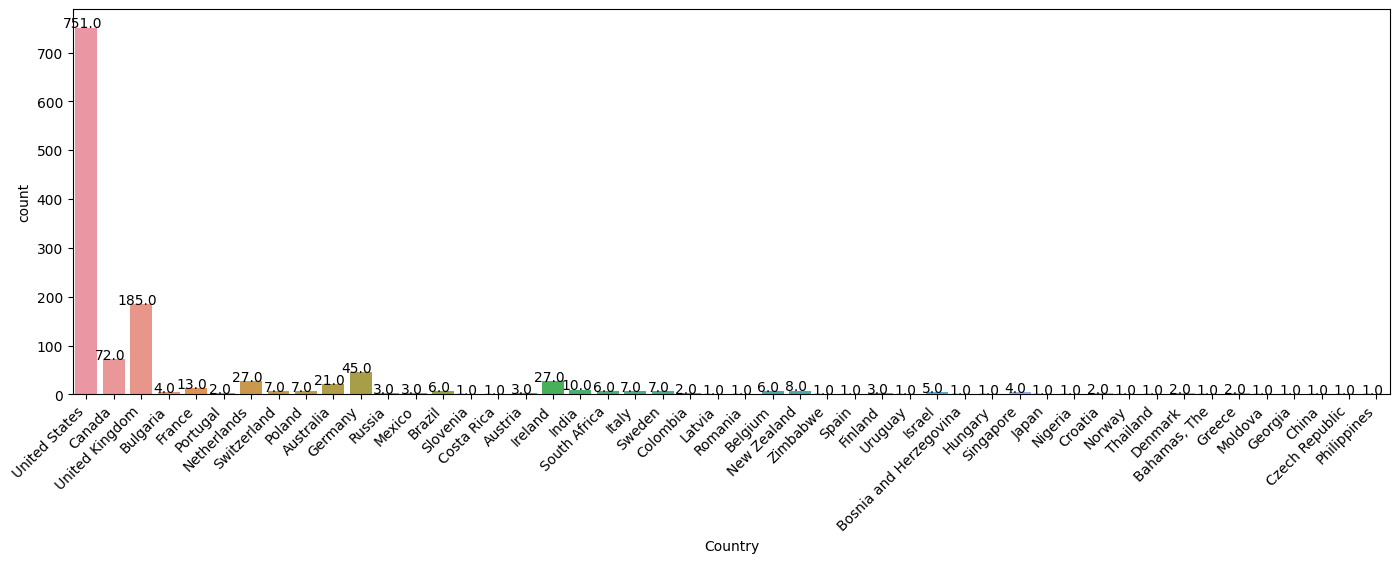
'wellness\_program', 'seek\_help', 'anonymity', 'leave',

'mental\_health\_consequence', 'phys\_health\_consequence', 'coworkers',

'supervisor', 'mental\_health\_interview', 'phys\_health\_interview',

'mental\_vs\_physical', 'obs\_consequence'],

dtype='object')



Min: -1726,

Max: 99999999999,

Mean: 79428148.31135821,

Median: 31.0

array([ 37, 44, 32, 31, 33,

35, 39, 42, 23, 29,

36, 27, 46, 41, 34,

30, 40, 38, 50, 24,

18, 28, 26, 22, 19,

25, 45, 21, -29, 43,

56, 60, 54, 329, 55,

99999999999, 48, 20, 57, 58,

47, 62, 51, 65, 49,

-1726, 5, 53, 61, 8,

11, -1, 72])

