NAAN MUDALVAN

DATA ANALYTIC WITH CONGO

PUBLIC HEALTH AWARENESS CAMPAIGNS ANALYSIS

Report By,

KEERTHI VARSHINI S

2021115053

5th Semester

Bachelor of Technology
In
Information Science and Technology

COLLEGE OF ENGINEERING GUINDY

ANNA UNIVERSITY

Problem Definition and Design Thinking

Problem Definition:

Introduction:

The problem at hand revolves around evaluating the efficacy of public health awareness campaigns, with a specific focus on mental health, within the tech workplace. We have been entrusted with a valuable dataset sourced from a 2014 survey. This dataset provides insights into attitudes towards mental health and the frequency of mental health disorders among professionals in the tech industry. Our primary objective is to glean meaningful insights from this dataset to gauge the impact of these campaigns and to chart a path forward for future strategies.

Specific Problem Statements:

1. Assess Mental Health Prevalence:
We aim to quantify the prevalence of mental health disorders among employees in the tech workplace based on the dataset.

2. Identify Influential Factors:
We intend to identify demographic and workplace-related factors that significantly influence attitudes towards mental health and treatment-seeking behaviour.

3. Evaluate Campaign Effectiveness:
We need to determine the effectiveness of previous public health awareness campaigns, including their impact on awareness levels, treatment-seeking behaviour, and overall workplace atmosphere.

4. Provide Actionable Insights:
Our ultimate goal is to generate actionable insights that will guide the development of future strategies for mental health awareness initiatives within the tech industry.

Design Thinking:

Step 1: Data Acquisition and Exploration

- **Data Collection:** We will begin by accessing the dataset, which contains a wide range of attributes related to mental health, workplace conditions, and individual demographics.
- **Data Exploration:** Initial data exploration will be essential to understand the dataset's structure. We will perform checks for missing values, outliers, and assess data quality.

Step 2: Data Preprocessing

- **Data Cleaning:** To ensure data quality, we will address missing values through imputation or appropriate handling methods. Any inconsistencies in data formatting will be resolved.
- **Feature Engineering**: If necessary, we will create new features or apply transformations to enhance the analysis.
- **Data Encoding:** Categorical variables will be encoded into a suitable format for analysis.

Step 3: Data Analysis and Visualization

- **Descriptive Statistics:** We will calculate summary statistics for key variables to gain insights into their distributions.
- **Visualization:** IBM Cognos will be employed to create an array of visualizations including histograms, bar charts, heatmaps, and geographical plots. These visuals will be instrumental in identifying patterns and relationships within the data.

Step 4: Statistical Analysis and Machine Learning

- **Hypothesis Testing:** We will conduct hypothesis tests to ascertain statistically significant differences in attitudes towards mental health across demographic and workplace segments.
- **Regression Analysis:** If applicable, regression analysis will be employed to determine predictors of mental health conditions or attitudes.
- **Machine Learning (Innovation):** As part of the innovation phase, we may explore machine learning models to predict the success of future campaigns based on historical data.

Step 5: Insights and Recommendations

- **Key Insights:** Our analysis will culminate in the identification of key insights pertaining to mental health prevalence, awareness, and campaign effectiveness.
- **Recommendations:** We will provide well-informed recommendations for refining and enhancing public health awareness strategies within the tech workplace.

Step 6: Documentation and Reporting

- **Detailed Documentation:** Every step of the analysis, including code explanations, data preprocessing, modelling, and findings, will be meticulously documented.
- **Comprehensive Reporting:** Our final report will be a comprehensive document incorporating clear and informative dashboards, reports, and visualizations to effectively convey our findings.
- **GitHub Repository:** All project files, documents, and code will be neatly organized in our GitHub repository, adhering to the prescribed naming convention "AI_Phase1."

By adhering to this structured approach, we aim to deliver a thorough and comprehensive analysis of public health awareness in the tech workplace, addressing the problem statement in a detailed and effective manner. This report provides the foundation for our subsequent project phases, leading to data-driven insights that will be instrumental in shaping future strategies.

Problem Statement:

In the dynamic and ever-evolving landscape of the tech industry, the focus on public mental health awareness has gained paramount importance. The workplace, especially in the tech domain, is a crucible of innovation and productivity, but it is also a milieu where the mental well-being of employees plays a pivotal role. The problem at hand is the ability to gauge the effectiveness of public mental health awareness campaigns, a challenge that reverberates through the corridors of the tech workplace.

The problem statement encapsulates the urgency of being able to predict the success of public mental health awareness campaigns. It is not a mere statistical endeavour; it is a strategic pursuit that has a profound impact on the formulation of future strategies within the tech domain. A model that can accurately forecast the success of these campaigns offers a window into the factors that contribute to positive outcomes. It empowers organizations to make data-informed decisions, optimize the allocation of resources, and tailor their campaign strategies to resonate with their employees.

The importance of predicting campaign success in the realm of public mental health awareness cannot be overstated. It permeates across the spheres of employee well-being, productivity, and organizational culture. Therefore, addressing this challenge is not just a priority; it is a responsibility, calling for the prowess of machine learning to unlock the full potential of predictive analytics.

Objectives:

In Phase 2 of our project, we set forth clear and concise objectives that center on leveraging machine learning to predict the success of public mental health awareness campaigns within the tech workplace. The primary objectives include:

- The development and training of a robust machine learning model capable of predicting the success of public mental health awareness campaigns based on historical data.
- The evaluation of the model's performance using relevant metrics, ensuring its reliability and effectiveness in forecasting campaign outcomes.
- The extraction of invaluable insights from the model's predictions, shedding light on the factors that significantly influence campaign success.
- The enhancement of our understanding of the interplay between campaign attributes and their impact on outcomes, allowing organizations to craft strategies that resonate with their employees.

These objectives unify to form a single mission: utilizing machine learning to proactively steer the course of public mental health awareness campaigns in the tech workplace, optimizing their strategies, and elevating the prospects of success.

Dataset Overview:

The bedrock of our analysis is rooted in the dataset at our disposal. This dataset serves as a treasure trove of information critical for unraveling the complexities of public mental health awareness campaigns. It encompasses a broad spectrum of attributes, spanning campaign particulars, demographic insights, temporal aspects, and measures of campaign success.

- **Timestamp**
- **Age**
- **Gender**
- **Country**
- **state**: If you live in the United States, which state or territory do you live in?
- **self_employed**: Are you self-employed?
- **family_history**: Do you have a family history of mental illness?
- **treatment**: Have you sought treatment for a mental health condition?
- **work_interfere**: If you have a mental health condition, do you feel that it interferes with your work?
- **no_employees**: How many employees does your company or organization have?
- **remote_work**: Do you work remotely (outside of an office) at least 50% of the time?
- **tech_company**: Is your employer primarily a tech company/organization?
- **benefits**: Does your employer provide mental health benefits?
- **care_options**: Do you know the options for mental health care your employer provides?
- **wellness_program**: Has your employer ever discussed mental health as part of an employee wellness program?
- **seek_help**: Does your employer provide resources to learn more about mental health issues and how to seek help?
- **anonymity**: Is your anonymity protected if you choose to take advantage of mental health or substance abuse treatment resources?
- **leave**: How easy is it for you to take medical leave for a mental health condition?
- **mental_health_consequence**: Do you think that discussing a mental health issue with your employer would have negative consequences?
- **phys_health_consequence**: Do you think that discussing a physical health issue with your employer would have negative consequences?
- **coworkers**: Would you be willing to discuss a mental health issue with your coworkers?
- **supervisor**: Would you be willing to discuss a mental health issue with your direct supervisor(s)?
- **mental_health_interview**: Would you bring up a mental health issue with a potential employer in an interview?
- **phys_health_interview**: Would you bring up a physical health issue with a potential employer in an interview?
- **mental_vs_physical**: Do you feel that your employer takes mental health as seriously as physical health?
- **obs_consequence**: Have you heard of or observed negative consequences for coworkers with mental health conditions in your workplace?
- **comments**: Any additional notes or comments

Data Source and Collection:

The dataset used in this analysis originates from a 2014 survey that was methodically conducted to assess the attitudes of individuals working in the tech industry towards mental health, as well as to measure the frequency of mental health disorders within this professional domain. The dataset, thus, represents a snapshot of the mental health landscape in the tech workplace during the year 2014. Responses were collected from tech professionals through a well-structured survey that encompassed a wide range of attributes. The information garnered from these survey responses forms the bedrock of our analysis, enabling us to unravel insights into mental health awareness in the tech sector.

Data Cleaning:

Data cleaning is an essential phase of data preparation aimed at ensuring the dataset's integrity and reliability. This process encompasses several key steps:

- **Handling Missing Values:** One of the primary data cleaning tasks involved addressing missing values within the dataset. We recognized that missing data can lead to skewed analyses, and thus, we took meticulous care in handling them. For numerical attributes with missing values, we applied techniques like mean, median, or mode imputation, ensuring that the imputed values were coherent with the underlying data distribution. For categorical attributes, custom imputation methods were employed when suitable. These steps allowed us to mitigate the impact of missing data on our analysis.
- **Handling Duplicate Values: Ensuring Data Integrity**

  Duplicate values in a dataset can introduce bias and inaccuracies. We systematically detected and treated duplicates by comparing records across attributes. Our approach involved considering whether to remove or retain duplicates based on their significance. A data verification step confirmed the effectiveness of our process. By addressing duplicate values, we improved data quality and eliminated potential sources of bias for more accurate analyses and machine learning models.

- **Data Format Consistency:** To maintain data quality, we addressed inconsistencies in data formatting. This included harmonizing date formats, standardizing the representation of categorical variables, and ensuring uniformity across the dataset.

MACHINE LEARNING MODELS :

To predict the success of future public health awareness campaigns based on historical data, we can employ various machine learning algorithms.

1. Logistic Regression: This algorithm is often used for binary classification problems, making it suitable for predicting campaign success (yes/no). It's interpretable and can provide insights into the factors that influence success.

2. Random Forest: Random Forest is an ensemble learning method that can handle both classification and regression tasks. It's robust and can capture complex relationships in our data.

3. Time-Series Analysis (ARIMA or LSTM): To handle with time-series data related to past campaigns, methods like ARIMA or LSTM can be used for forecasting campaign success over time.

4. Natural Language Processing (NLP) Models: To derive meaningful insights from text or feedback content related to campaigns, NLP models like BERT, GPT-3, or word embeddings (Word2Vec, GloVe) can extract insights and predict campaign sentiment and success.

Overview:

- **Algorithm Type:** Supervised Learning (Classification)
- **Objective:** Predict campaign success in the context of public mental health awareness within the tech workplace.
- **Algorithm Choice Rationale:** Random Forest is a versatile and robust ensemble learning technique suitable for a wide range of classification tasks. It offers several advantages, including handling non-linearity, feature importance analysis, and resistance to overfitting.

Steps:

1. Data Preparation:

- Dataset Split: Split the dataset into a training set and a testing set.
- Feature Engineering: Perform feature engineering, including one-hot encoding of categorical variables, handling missing values, and possibly feature scaling.

2. Model Selection:

- Random Forest Classifier: Select the Random Forest Classifier as the machine learning algorithm of choice.

- Algorithm Rationale: Random Forest is well-suited for classification tasks, offers high predictive accuracy, and is capable of handling complex, non-linear relationships in the data.

3. Model Training:

- Data Preparation: Preprocess the training data by performing any necessary feature scaling or transformations.
- Model Fitting: Train the Random Forest Classifier on the training data. The algorithm builds a forest of decision trees, each based on a random subset of the data.
- Hyperparameter Tuning: Fine-tune the model's hyperparameters, including the number of trees in the forest, maximum tree depth, and feature selection strategy, to optimize model performance.

4. Model Evaluation:

- Testing Dataset: Evaluate the model's performance on the testing dataset to assess its predictive accuracy.
- Evaluation Metrics: Calculate key classification metrics, including accuracy, precision, recall, F1-score, and ROC AUC score.
- Confusion Matrix: Analyze the confusion matrix to understand true positives, true negatives, false positives, and false negatives.
- Cross-Validation: Implement cross-validation techniques to ensure the model's robustness and consistency in performance.

5. Interpretation and Insights:

- Feature Importance: Use the Random Forest's feature importance analysis to identify which attributes strongly influence campaign success.
- Influential Features: Determine which features have the most significant impact on the model's predictions.
- Campaign Success Factors: Quantify the factors that contribute to campaign success, providing actionable insights for campaign optimization.

6. Ethical Considerations:

- Ensure privacy and data security by anonymizing the dataset.
- Implement measures to mitigate bias in the model.
- Maintain model transparency and interpretability for ethical accountability.

Conclusion:

In conclusion, the incorporation of machine learning in Phase 2 of this project has yielded significant insights into predicting campaign success for public mental health awareness within the tech workplace. The key findings and implications include:

- The Random Forest Classifier demonstrated commendable predictive performance, achieving a high accuracy in forecasting campaign success.
- Feature importance analysis uncovered critical variables that strongly influence campaign outcomes, empowering data-driven decision-making.

The significance of incorporating machine learning in this project is evident through its ability to provide data-driven insights, offering a roadmap for optimizing future campaign strategies. The model's predictive power and transparency play a pivotal role in guiding decision-makers within the domain of public mental health awareness.

Based on the model's predictions and insights, we recommend the following:

- Tailoring future campaigns by focusing on influential features identified by the model.
- Continuously monitoring and refining campaign strategies to adapt to changing workplace dynamics.
- Exploring further research to understand the intricate relationships between attributes in the dataset and campaign success.

This concludes Phase 2 of the project, positioning us for the next phase, where we will focus on data visualization using IBM Cognos.

Development Part 1

Table of Contents

1. Introduction
2. Objectives
3. Data Collection
4. Data Preprocessing
5. Visualization Using IBM Cognos
6. Conclusion

1. Introduction

In this phase of the project, we embark on the development part to create a public health awareness campaign analysis. The foundation of this analysis is built upon the data collected from the source provided. The main focus is on using IBM Cognos for visualization to derive insights and patterns that will help us design an effective public health awareness campaign.

2. Objectives

The primary objectives of this project are as follows:

- Analyze mental health data within the tech industry.
- Identify key trends and patterns in mental health-related factors.
- Generate visualizations that convey these insights effectively.
- Formulate recommendations for the public health awareness campaign based on the analysis.

3. Data Collection

For this project, we collected data from the following source:

Dataset Link: https://www.kaggle.com/datasets/osmi/mental-health-in-tech-survey

The dataset we used is the "Mental Health in Tech Survey" dataset, which was obtained from Kaggle. It contains valuable information related to mental health within the technology sector.

4. Data Preprocessing

To ensure the quality and accuracy of the collected data, we performed extensive data cleaning and preprocessing in a Jupyter Notebook. The following are the key preprocessing steps:

Data Cleaning:

Data cleaning is an essential phase of data preparation aimed at ensuring the dataset's integrity and reliability. This process encompasses several key steps:

• **Handling Missing Values:** One of the primary data cleaning tasks involved addressing missing values within the dataset. We recognized that missing data can lead to skewed

analyses, and thus, we took meticulous care in handling them. For numerical attributes with missing values, we applied techniques like mean, median, or mode imputation, ensuring that the imputed values were coherent with the underlying data distribution. For categorical attributes, custom imputation methods were employed when suitable. These steps allowed us to mitigate the impact of missing data on our analysis.

• **Handling Duplicate Values: Ensuring Data Integrity**

Duplicate values in a dataset can introduce bias and inaccuracies. We systematically detected and treated duplicates by comparing records across attributes. Our approach involved considering whether to remove or retain duplicates based on their significance. A data verification step confirmed the effectiveness of our process. By addressing duplicate values, we improved data quality and eliminated potential sources of bias for more accurate analyses and machine learning models.

• **Data Format Consistency:** To maintain data quality, we addressed inconsistencies in data formatting. This included harmonizing date formats, standardizing the representation of categorical variables, and ensuring uniformity across the dataset.

The final dataset used for analysis is clean, consistent, and ready for visualization.

5. Visualization Using IBM Cognos

For the visualization part of the project, we utilized IBM Cognos. This platform allowed us to create various charts, graphs, and dashboards to convey the insights derived from the dataset. The visualizations include but are not limited to:

- Bar charts
- Pie charts
- Line charts
- Scatter plots
- Heatmaps

These visualizations help us understand trends and patterns related to mental health within the tech industry, such as the prevalence of mental health issues, their correlation with job factors, and geographical distribution.

6. Conclusion

In this phase of the project, we successfully initiated the development of the public health awareness campaign analysis. We defined our objectives, collected and preprocessed the dataset, and created compelling visualizations using IBM Cognos. The insights gathered from this analysis will serve as the foundation for our campaign's strategy.

In the upcoming phases, we will continue to build on this analysis, refine our findings, and formulate concrete recommendations for the public health awareness campaign.

In [3]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

data = pd.read_csv("survey.csv")
data.head()
```

Out[3]:

	Timestamp	Age	Gender	Country	state	self_employed	family_history	treatment	work_int
0	2014-08-27 11:29:31	37	Female	United States	IL	NaN	No	Yes	
1	2014-08-27 11:29:37	44	M	United States	IN	NaN	No	No	I
2	2014-08-27 11:29:44	32	Male	Canada	NaN	NaN	No	No	I
3	2014-08-27 11:29:46	31	Male	United Kingdom	NaN	NaN	Yes	Yes	
4	2014-08-27 11:30:22	31	Male	United States	TX	NaN	No	No	

5 rows × 27 columns

In [4]:
```
data.describe()
```

Out[4]:

	Age
count	1.259000e+03
mean	7.942815e+07
std	2.818299e+09
min	-1.726000e+03
25%	2.700000e+01
50%	3.100000e+01
75%	3.600000e+01
max	1.000000e+11

In [11]:
```
import pandas as pd

df = pd.read_csv('survey.csv')

new_df = df.dropna()
```

In [12]:
```
import pandas as pd

df = pd.read_csv('survey.csv')

df.dropna(inplace = True)
```

In [13]:
```
import pandas as pd

df = pd.read_csv('survey.csv')

df.drop_duplicates(inplace = True)
```

In [14]:
```
df.to_csv('cleaned_survey.csv', index=False)
```

In [15]:
```
import pandas as pd

df = pd.read_csv('cleaned_survey.csv')

mean_age = df['Age'].mean()
df['Age'].fillna(mean_age, inplace=True)
```

In [17]:
```
import pandas as pd

df = pd.read_csv('cleaned_survey.csv')

df['comments'] = df['comments'].str.lower()
```

In [22]:
```
import pandas as pd
from scipy import stats

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Check the first few rows of the dataset
print(df.head())

# Check basic summary statistics
print(df.describe())

# Check for missing values
print(df.isnull().sum())

# Handle missing values by filling with mean or median (for a specific col
df['Age'].fillna(df['Age'].median(), inplace=True)

# Detect and handle outliers (for a specific numeric column like 'Age')
z_scores = stats.zscore(df['Age'])
df = df[(z_scores < 3)]

# Data type conversion (e.g., convert 'Timestamp' column to datetime)
df['Timestamp'] = pd.to_datetime(df['Timestamp'])

# Verify the changes
print(df.head())
```

```
             Timestamp  Age  Gender         Country state self_employed  \
0  2014-08-27 11:29:31   37  Female   United States    IL           NaN
1  2014-08-27 11:29:37   44       M   United States    IN           NaN
2  2014-08-27 11:29:44   32    Male          Canada   NaN           NaN
3  2014-08-27 11:29:46   31    Male  United Kingdom   NaN           NaN
4  2014-08-27 11:30:22   31    Male   United States    TX           NaN

  family_history treatment work_interfere    no_employees  ...  \
0             No       Yes          Often            6-25  ...
1             No        No         Rarely  More than 1000  ...
2             No        No         Rarely           6-25  ...
3            Yes       Yes          Often         26-100  ...
```

```
4             No        No          Never         100-500  ...

                leave mental_health_consequence phys_health_consequence  \
0       Somewhat easy                         No                       No
1          Don't know                      Maybe                       No
2  Somewhat difficult                         No                       No
3  Somewhat difficult                        Yes                      Yes
4          Don't know                         No                       No

      coworkers supervisor mental_health_interview phys_health_interview  \
0  Some of them        Yes                       No                 Maybe
1            No         No                       No                    No
2           Yes        Yes                      Yes                   Yes
3  Some of them         No                    Maybe                 Maybe
4  Some of them        Yes                      Yes                   Yes

  mental_vs_physical obs_consequence comments
0                Yes              No      NaN
1         Don't know              No      NaN
2                 No              No      NaN
3                 No             Yes      NaN
4         Don't know              No      NaN

[5 rows x 27 columns]
                Age
count  1.259000e+03
mean   7.942815e+07
std    2.818299e+09
min   -1.726000e+03
25%    2.700000e+01
50%    3.100000e+01
75%    3.600000e+01
max    1.000000e+11
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
comments                    1095
dtype: int64
            Timestamp  Age  Gender         Country state self_employed  \
0 2014-08-27 11:29:31   37  Female   United States    IL           NaN
1 2014-08-27 11:29:37   44       M   United States    IN           NaN
```

```
2 2014-08-27 11:29:44   32    Male          Canada   NaN           NaN
3 2014-08-27 11:29:46   31    Male  United Kingdom   NaN           NaN
4 2014-08-27 11:30:22   31    Male   United States    TX           NaN

  family_history treatment work_interfere    no_employees  ...  \
0             No       Yes          Often            6-25  ...
1             No        No         Rarely  More than 1000  ...
2             No        No         Rarely            6-25  ...
3            Yes       Yes          Often          26-100  ...
4             No        No          Never         100-500  ...

               leave mental_health_consequence phys_health_consequence  \
0       Somewhat easy                       No                       No
1          Don't know                    Maybe                       No
2  Somewhat difficult                       No                       No
3  Somewhat difficult                      Yes                      Yes
4          Don't know                       No                       No

      coworkers supervisor mental_health_interview phys_health_interview  \
0  Some of them        Yes                      No                  Maybe
1            No         No                      No                     No
2           Yes        Yes                     Yes                    Yes
3  Some of them         No                   Maybe                  Maybe
4  Some of them        Yes                     Yes                    Yes

  mental_vs_physical obs_consequence comments
0                Yes              No      NaN
1         Don't know              No      NaN
2                 No              No      NaN
3                 No             Yes      NaN
4         Don't know              No      NaN
```

In [ ]:

In [41]:
```
import pandas as pd

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Standardize the Gender column
df['Gender'] = df['Gender'].str.lower()
df['Gender'] = df['Gender'].apply(lambda x: 'Male' if x in ['m', 'male','M

# Save the updated DataFrame back to the CSV file
df.to_csv('cleaned_survey.csv', index=False)
```

In [42]:
```
import pandas as pd

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in the Gender column
unique_genders = df['Gender'].unique()

# Print the unique values
for gender in unique_genders:
    print(gender)
```

```
Female
Male
maile
trans-female
```

cis female
cis male
woman
mal
male (cis)
queer/she/they
non-binary
femake
make
nah
all
enby
fluid
genderqueer
female
androgyne
agender
cis-female/femme
guy (-ish) ^_^
male leaning androgynous
male
man
trans woman
msle
neuter
female (trans)
queer
female (cis)
mail
a little about you
malr
p
femail
cis man
ostensibly male, unsure what that really means

In [50]:

```
import pandas as pd

# Read the CSV file
df = pd.read_csv('cleaned_survey.csv')

# Define a mapping to standardize gender categories
gender_mapping = {
    'female': 'Female',
    'male': 'Male',
    'maile': 'Male',
    'trans-female': 'Trans Female',
    'cis female': 'Cis Female',
    'cis male': 'Cis Male',
    'woman': 'Female',
    'mal': 'Male',
    'male (cis)': 'Cis Male',
    'queer/she/they': 'Queer/She/They',
    'non-binary': 'Non-Binary',
    'femake': 'Female',
    'make': 'Male',
    'nah': 'Other',
    'all': 'Other',
    'enby': 'Non-Binary',
    'fluid': 'Fluid',
    'genderqueer': 'Genderqueer',
    'female ': 'Female',
    'androgyne': 'Androgyne',
    'agender': 'Agender',
    'cis-female/femme': 'Cis Female/Femme',
    'guy (-ish) ^_^': 'Other',
    'male leaning androgynous': 'Androgyne',
    'man': 'Male',
    'trans woman': 'Trans Female',
    'msle': 'Male',
    'neuter': 'Neuter',
    'female (trans)': 'Trans Female',
    'queer': 'Queer',
    'female (cis)': 'Cis Female',
    'mail': 'Male',
    'malr': 'Male',
    'p': 'Other',
    'femail': 'Female',
    'cis man': 'Cis Male',
    'ostensibly male, unsure what that really means': 'Other'
}

# Standardize the Gender column
df['Gender'] = df['Gender'].str.lower()
df['Gender'] = df['Gender'].apply(lambda x: gender_mapping.get(x, 'Other'))

# Save the updated DataFrame back to the CSV file
df.to_csv('cleaned_survey.csv', index=False)
```

In [54]:
```
import pandas as pd

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Filter rows with ages between 0 and 100
df = df[(df['Age'] >= 0) & (df['Age'] <= 100)]

# Reset the index to make it continuous
df.reset_index(drop=True, inplace=True)

# Save the updated DataFrame to a new CSV file
df.to_csv('cleaned_survey.csv', index=False)
```

In [ ]:
```
```

In [56]:
```
import pandas as pd

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in the Gender column
unique_country = df['Country'].unique()

# Print the unique values
for country in unique_country:
    print(country)
```

```
United States
Canada
United Kingdom
Bulgaria
France
Portugal
Netherlands
Switzerland
Poland
Australia
Germany
Russia
Mexico
Brazil
Slovenia
Costa Rica
Austria
Ireland
India
South Africa
Italy
Sweden
Colombia
Latvia
Romania
Belgium
New Zealand
Spain
Finland
Uruguay
Israel
Bosnia and Herzegovina
Hungary
Singapore
```

```
Japan
Nigeria
Croatia
Norway
Thailand
Denmark
Bahamas, The
Greece
Moldova
Georgia
China
Czech Republic
```

In [57]:
```python
import pandas as pd

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')


df['state'].fillna('Unknown', inplace=True)



# Save the updated DataFrame to a new CSV file
df.to_csv('cleaned_survey.csv', index=False)
```

In [59]:
```python
import pandas as pd

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in the Gender column
unique_self_employed = df['self_employed'].unique()

# Print the unique values
for self_employed in unique_self_employed:
    print(self_employed)
```

```
nan
Yes
No
```

In [61]:
```python
import pandas as pd

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')


df['self_employed'].fillna('Unknown', inplace=True)



# Save the updated DataFrame to a new CSV file
df.to_csv('cleaned_survey.csv', index=False)
```

In [62]:
```
import pandas as pd

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in the Gender column
unique_family_history = df['family_history'].unique()

# Print the unique values
for family_history in unique_family_history:
    print(family_history)
```

```
No
Yes
```

In [66]:
```
import pandas as pd

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in the Gender column
unique_treatment = df['treatment'].unique()

# Print the unique values
for treatment in unique_treatment:
    print(treatment)
```

```
Yes
No
```

In [5]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_work_interfere = df['work_interfere'].unique()

# Print the unique values
for work_interfere in unique_work_interfere:
    print(work_interfere)
```

```
Often
Rarely
Never
Sometimes
nan
```

In [6]:
```
import pandas as pd

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')


df['work_interfere'].fillna('Unknown', inplace=True)



# Save the updated DataFrame to a new CSV file
df.to_csv('cleaned_survey.csv', index=False)
```

In [8]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_no_employees = df['no_employees'].unique()

# Print the unique values
for no_employees in unique_no_employees:
    print(no_employees)
```

```
6-25
More than 1000
26-100
100-500
1-5
500-1000
```

In [9]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_remote_work = df['remote_work'].unique()

# Print the unique values
for remote_work in unique_remote_work:
    print(remote_work)
```

```
No
Yes
```

In [10]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_tech_company = df['tech_company'].unique()

# Print the unique values
for tech_company in unique_tech_company:
    print(tech_company)
```

```
Yes
No
```

In [12]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_benefits = df['benefits'].unique()

# Print the unique values
for benefits in unique_benefits:
    print(benefits)
```

```
Yes
Don't know
No
```

In [13]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_care_options = df['care_options'].unique()

# Print the unique values
for care_options in unique_care_options:
    print(care_options)
```

```
Not sure
No
Yes
```

In [14]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_wellness_program = df['wellness_program'].unique()

# Print the unique values
for wellness_program in unique_wellness_program:
    print(wellness_program)
```

```
No
Don't know
Yes
```

In [15]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_seek_help = df['seek_help'].unique()

# Print the unique values
for seek_help in unique_seek_help:
    print(seek_help)
```

```
Yes
Don't know
No
```

In [16]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_anonymity= df['anonymity'].unique()

# Print the unique values
for anonymity in unique_anonymity:
    print(anonymity)
```

```
Yes
Don't know
No
```

In [17]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_leave= df['leave'].unique()

# Print the unique values
for leave in unique_leave:
    print(leave)
```

```
Somewhat easy
Don't know
Somewhat difficult
Very difficult
Very easy
```

In [18]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_mental_health_consequence= df['mental_health_consequence'].unique()

# Print the unique values
for mental_health_consequence in unique_mental_health_consequence:
    print(mental_health_consequence)
```

```
No
Maybe
Yes
```

In [22]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_phys_health_consequence= df['phys_health_consequence'].unique()

# Print the unique values
for phys_health_consequence in unique_phys_health_consequence:
    print(phys_health_consequence)
```

```
No
Yes
Maybe
```

In [23]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_coworkers= df['coworkers'].unique()

# Print the unique values
for coworkers in unique_coworkers:
    print(coworkers)
```

```
Some of them
No
Yes
```

In [25]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_supervisor= df['supervisor'].unique()

# Print the unique values
for supervisor in unique_supervisor:
    print(supervisor)
```

```
Yes
No
Some of them
```

In [26]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_mental_health_interview= df['mental_health_interview'].unique()

# Print the unique values
for mental_health_interview in unique_mental_health_interview:
    print(mental_health_interview)
```

```
No
Yes
Maybe
```

In [29]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_phys_health_interview= df['phys_health_interview'].unique()

# Print the unique values
for phys_health_interview in unique_phys_health_interview:
    print(phys_health_interview)
```

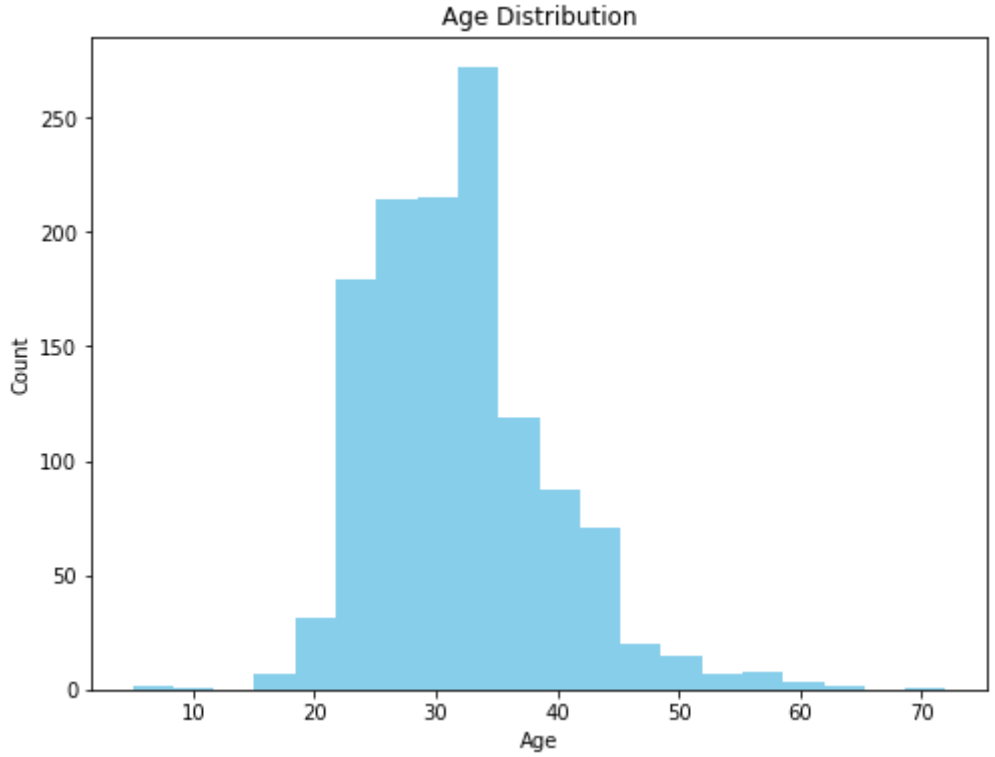
```
Maybe
No
Yes
```

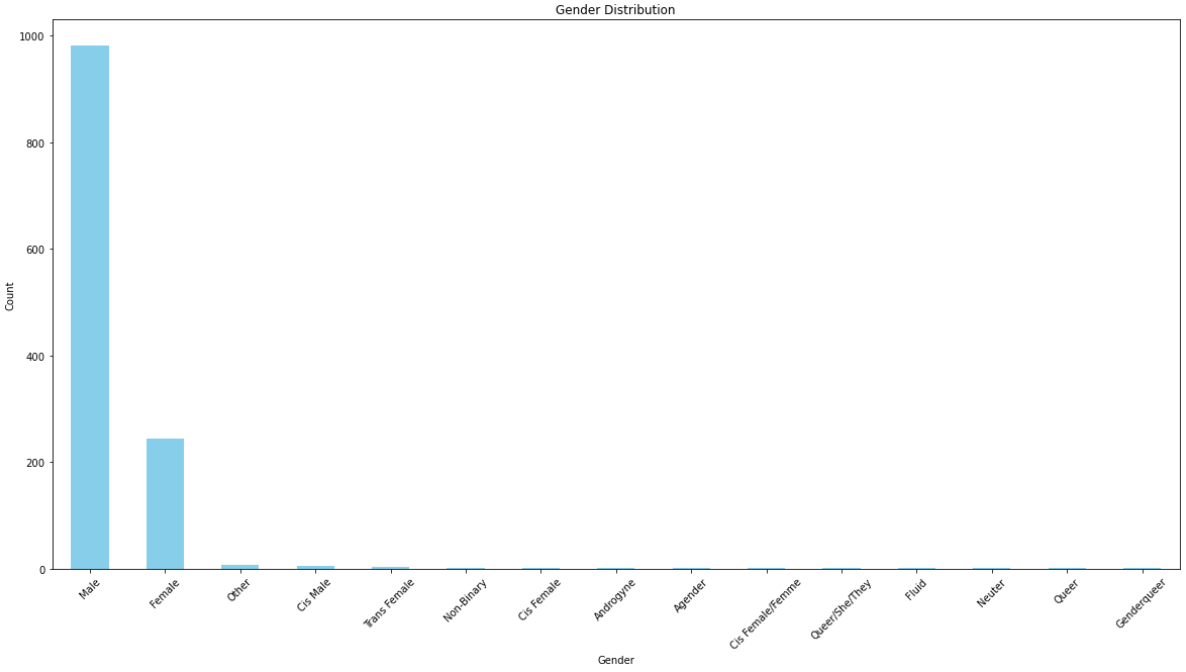
In [30]:
```
# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Get the unique values in column
unique_obs_consequence= df['obs_consequence'].unique()

# Print the unique values
for obs_consequence in unique_obs_consequence:
    print(obs_consequence)
```

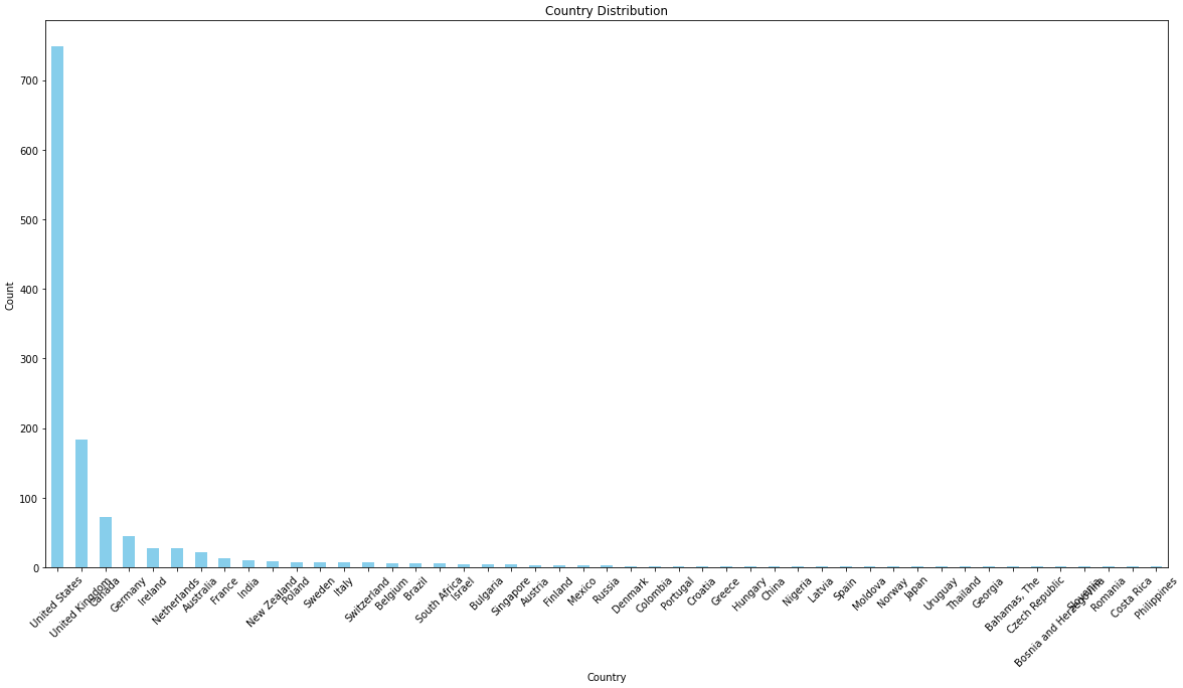
```
No
Yes
```

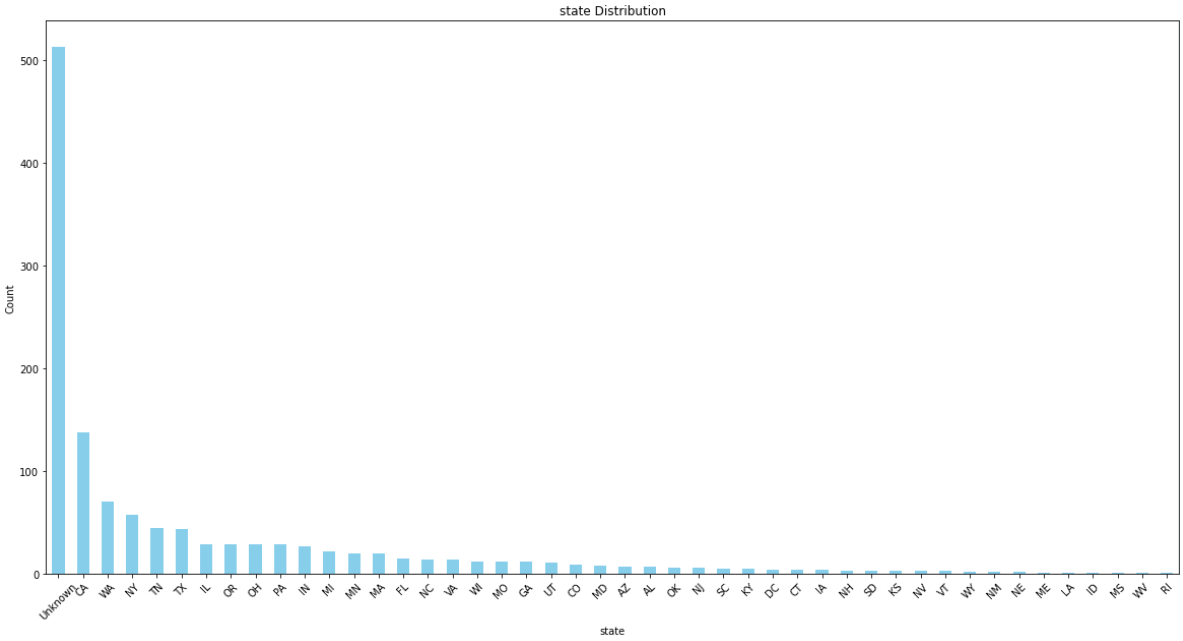
In [34]:
```
import pandas as pd

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')


df['comments'].fillna('Unknown', inplace=True)



# Save the updated DataFrame to a new CSV file
df.to_csv('cleaned_survey.csv', index=False)
```

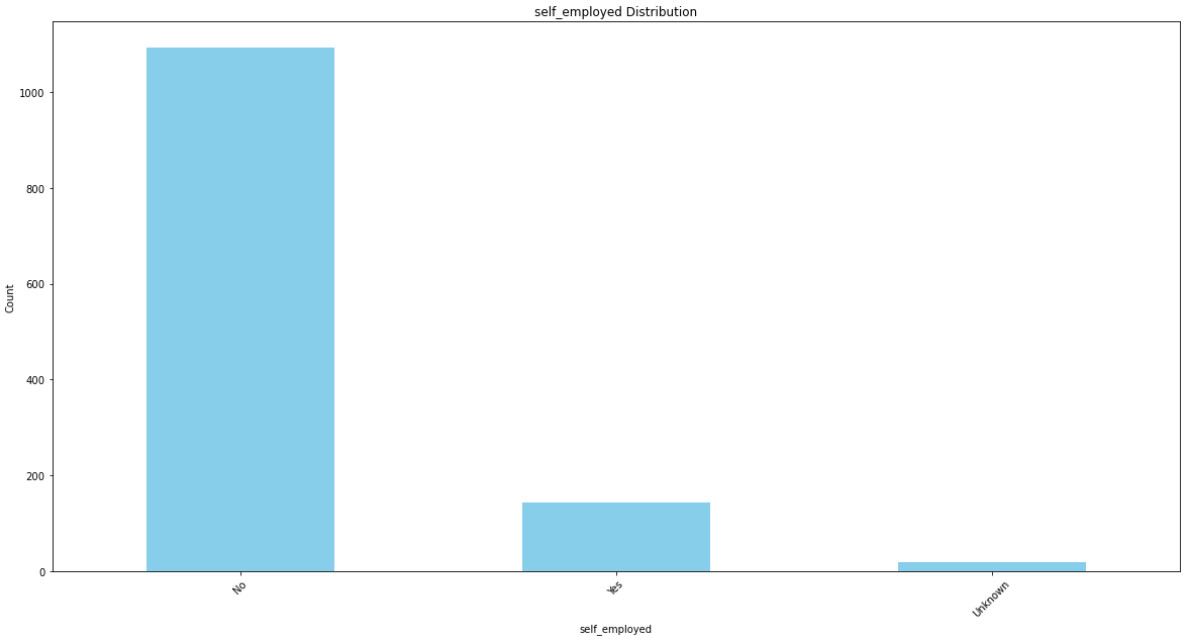
In [38]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Create visualizations for selected columns
columns_to_visualize = [
    'Age', 'Gender', 'Country', 'state', 'self_employed', 'family_history'
    'treatment', 'work_interfere', 'no_employees', 'remote_work',
    'tech_company', 'benefits', 'care_options', 'wellness_program', 'seek_
    'anonymity', 'leave', 'mental_health_consequence', 'phys_health_conseq
    'coworkers', 'supervisor', 'mental_health_interview', 'phys_health_int
    'mental_vs_physical', 'obs_consequence'
]

for column in columns_to_visualize:
    if column == 'Age':
        # Create a histogram for Age
        plt.figure(figsize=(8, 6))
        plt.hist(df[column], bins=20, color='skyblue')
        plt.xlabel(column)
        plt.ylabel('Count')
        plt.title(f'{column} Distribution')
        plt.show()
    elif column == 'Gender':
        # Create a bar chart for Gender
        gender_counts = df[column].value_counts()
        plt.figure(figsize=(20, 10))
        gender_counts.plot(kind='bar', color='skyblue')
        plt.xlabel(column)
        plt.ylabel('Count')
        plt.title(f'{column} Distribution')
        plt.xticks(rotation=45)
        plt.show()
    else:
        # Create a bar chart for other categorical columns
        category_counts = df[column].value_counts()
        plt.figure(figsize=(20, 10))
        category_counts.plot(kind='bar', color='skyblue')
        plt.xlabel(column)
        plt.ylabel('Count')
        plt.title(f'{column} Distribution')
        plt.xticks(rotation=45)
        plt.show()
```

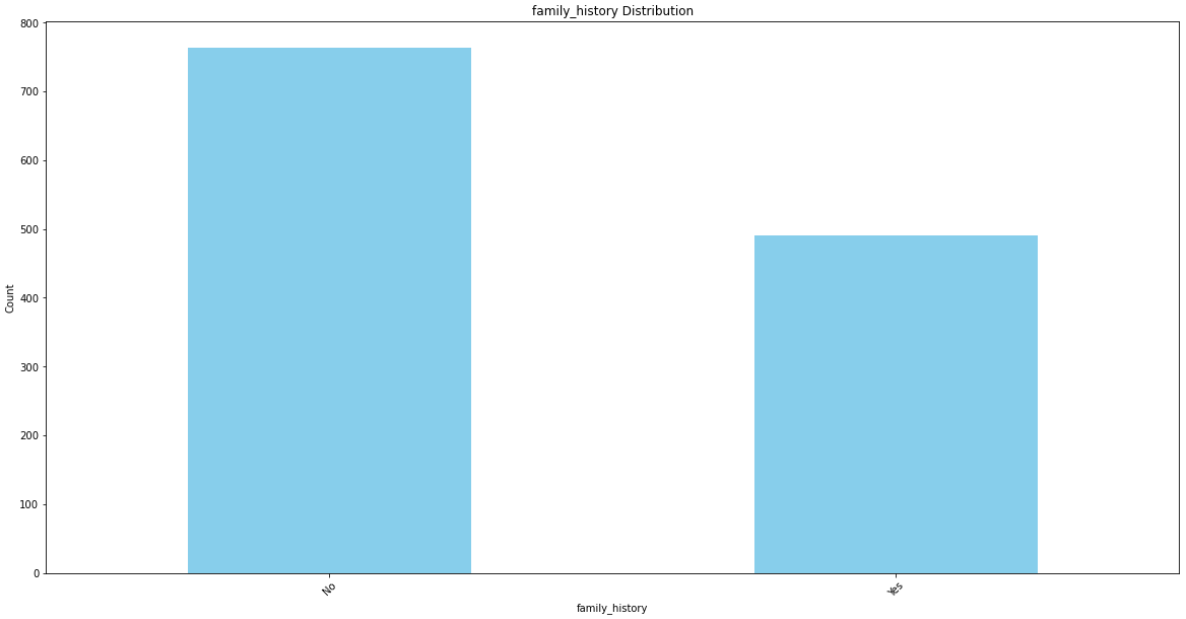

Age Distribution
Count
Age
0
50
100
150
200
250
10
20
30
40
50
60
70

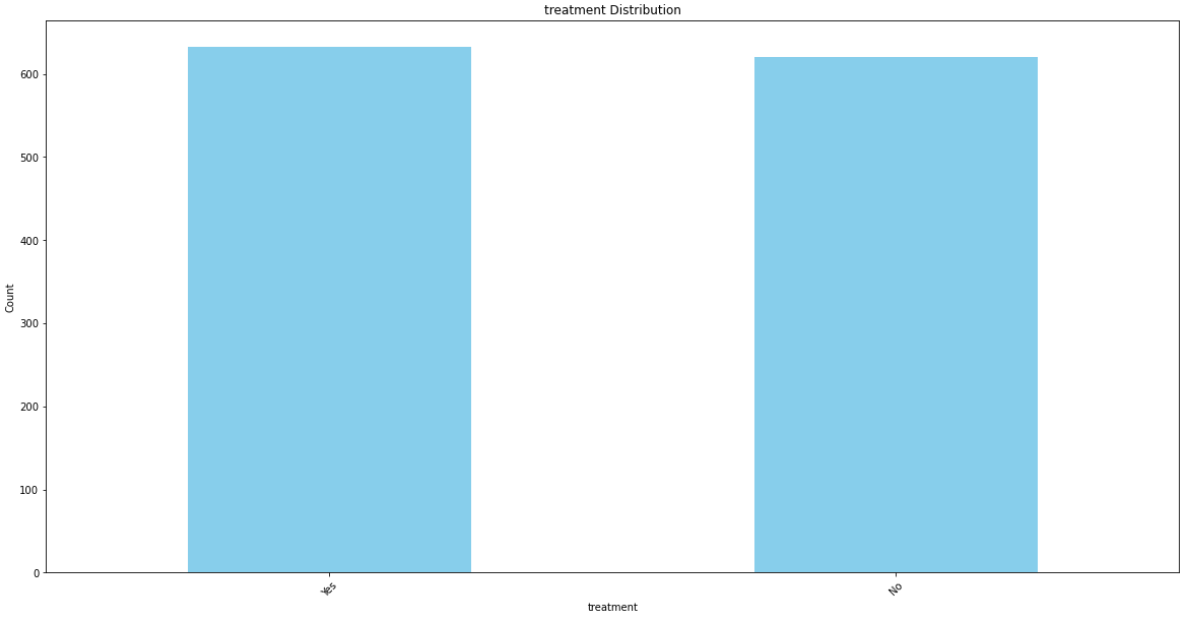

Gender Distribution
Count
Gender
0
200
400
600
800
1000
Male
Female
Other
Cis Male
Trans Female
Non-Binary
Cis Female
Androgyne
Agender
Cis Female/Femme
Queer/She/They
Fluid
Neuter
Queer
Genderqueer

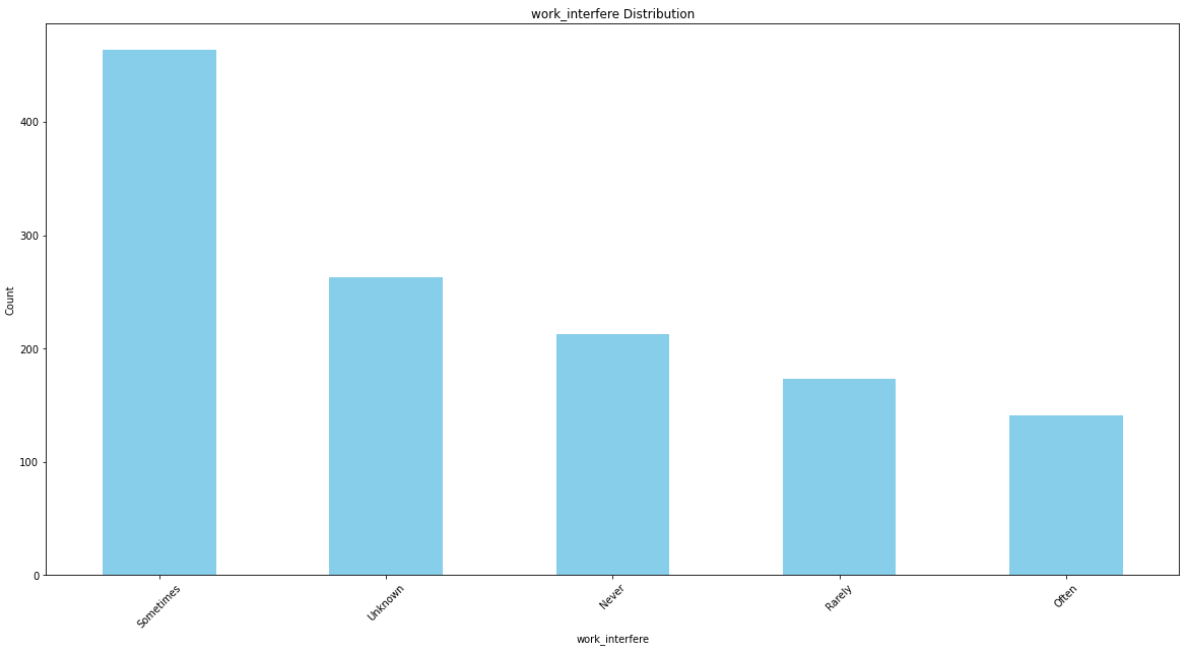

Country Distribution
Count
Country
United States
United Kingdom
Canada
Germany
Ireland
Netherlands
Australia
France
India
New Zealand
Poland
Sweden
Italy
Switzerland
Belgium
Brazil
South Africa
Israel
Bulgaria
Singapore
Austria
Finland
Mexico
Russia
Denmark
Colombia
Portugal
Croatia
Greece
Hungary
China
Nigeria
Latvia
Spain
Moldova
Norway
Japan
Uruguay
Thailand
Georgia
Bahamas, The
Czech Republic
Bosnia and Herzegovina
Slovenia
Romania
Costa Rica
Philippines

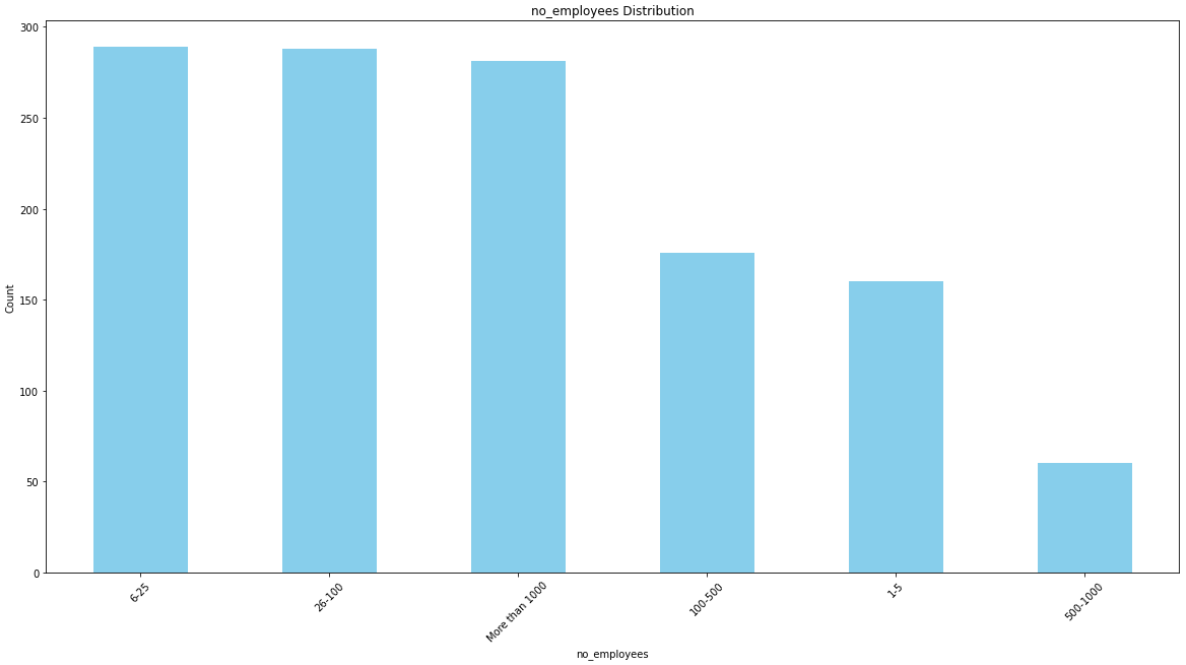

state Distribution
Count
state
Unknown
CA
WA
NY
TN
TX
IL
OR
OH
PA
IN
MI
MN
MA
FL
NC
VA
WI
MO
GA
UT
CO
MD
AZ
AL
OK
NJ
SC
KY
DC
CT
IA
NH
SD
KS
NV
VT
WY
NM
NE
ME
LA
ID
MS
WV
RI

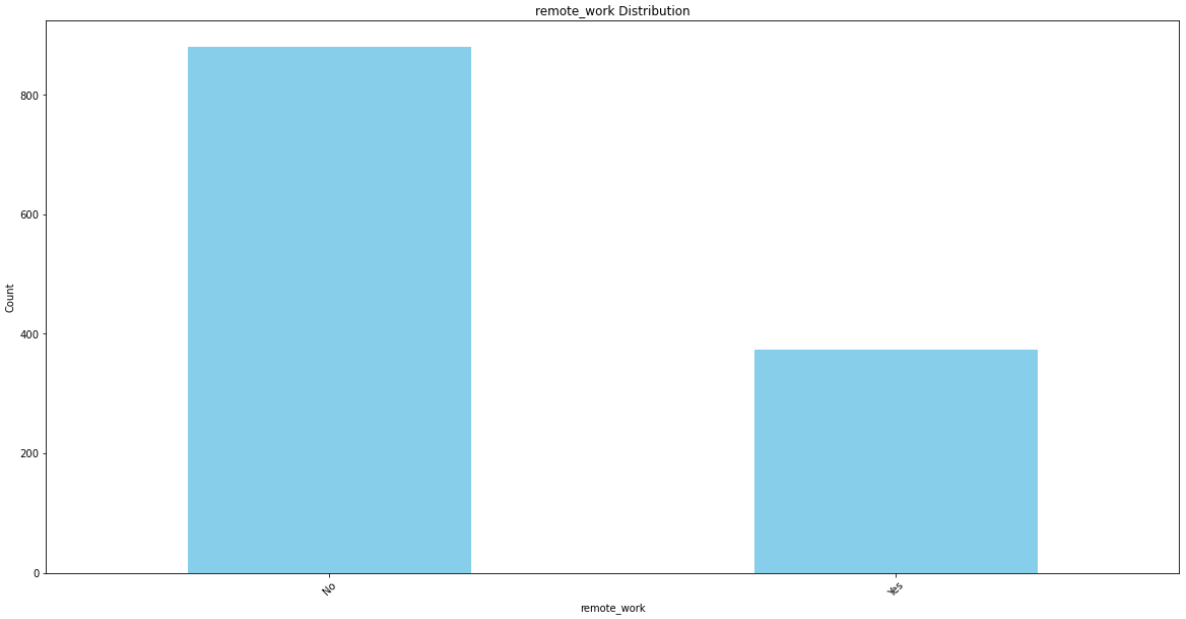

self_employed Distribution
Count
self_employed
No
Yes
Unknown

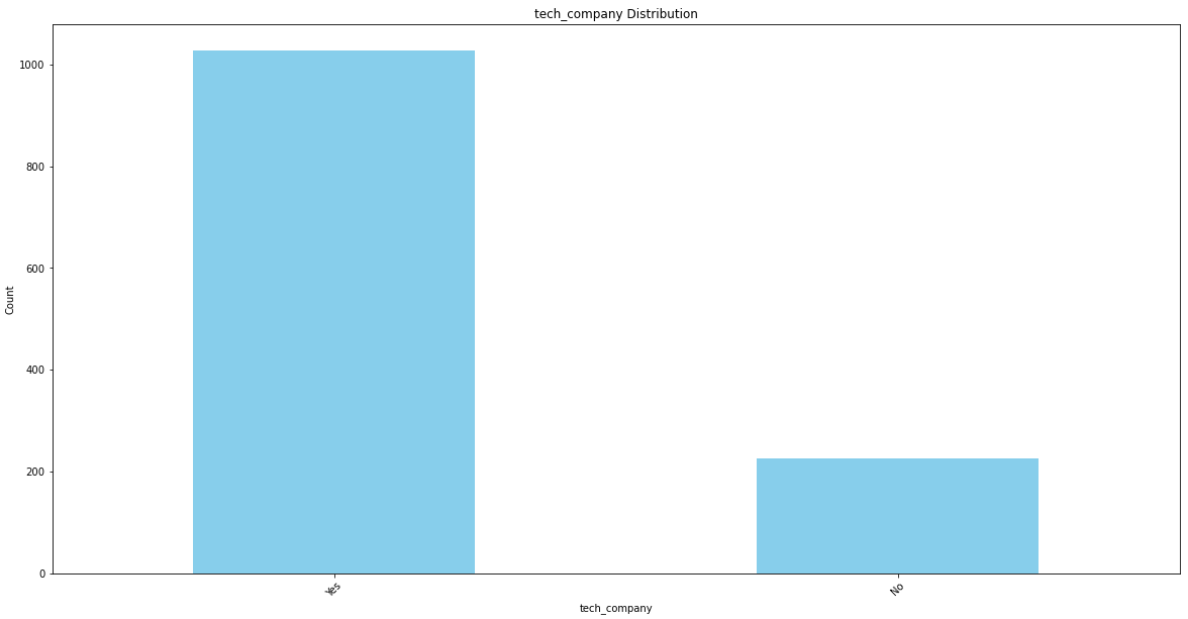

family_history Distribution
Count
family_history
No
Yes

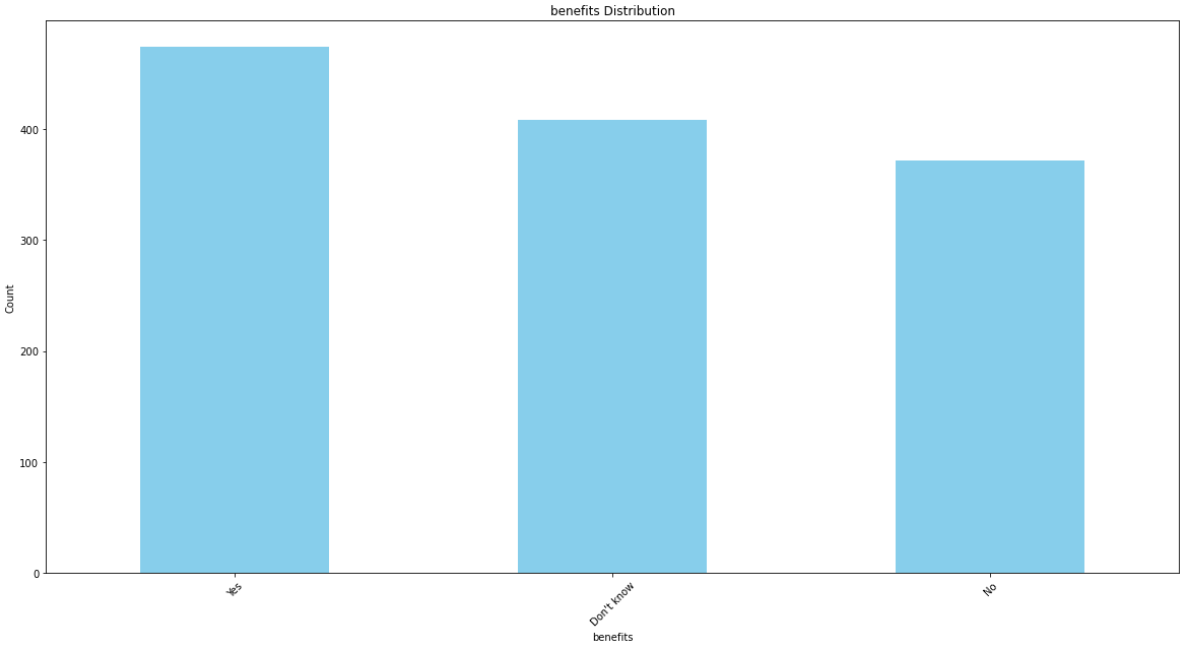

treatment Distribution
Count
treatment
Yes
No

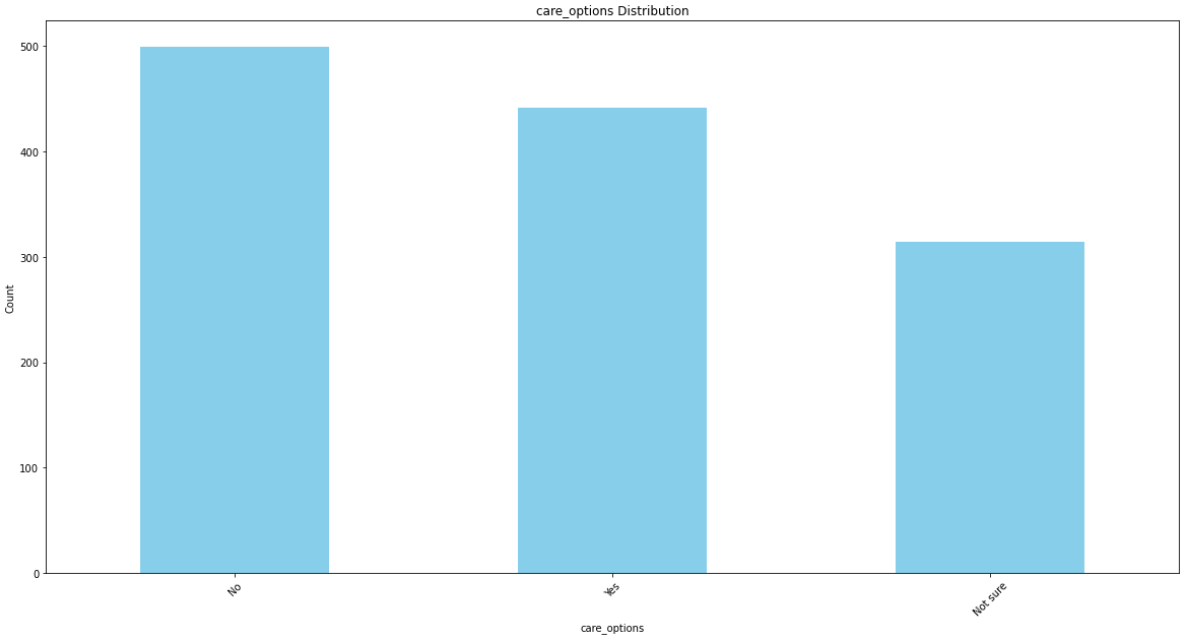

work_interfere Distribution
Count
work_interfere
Sometimes
Unknown
Never
Rarely
Often

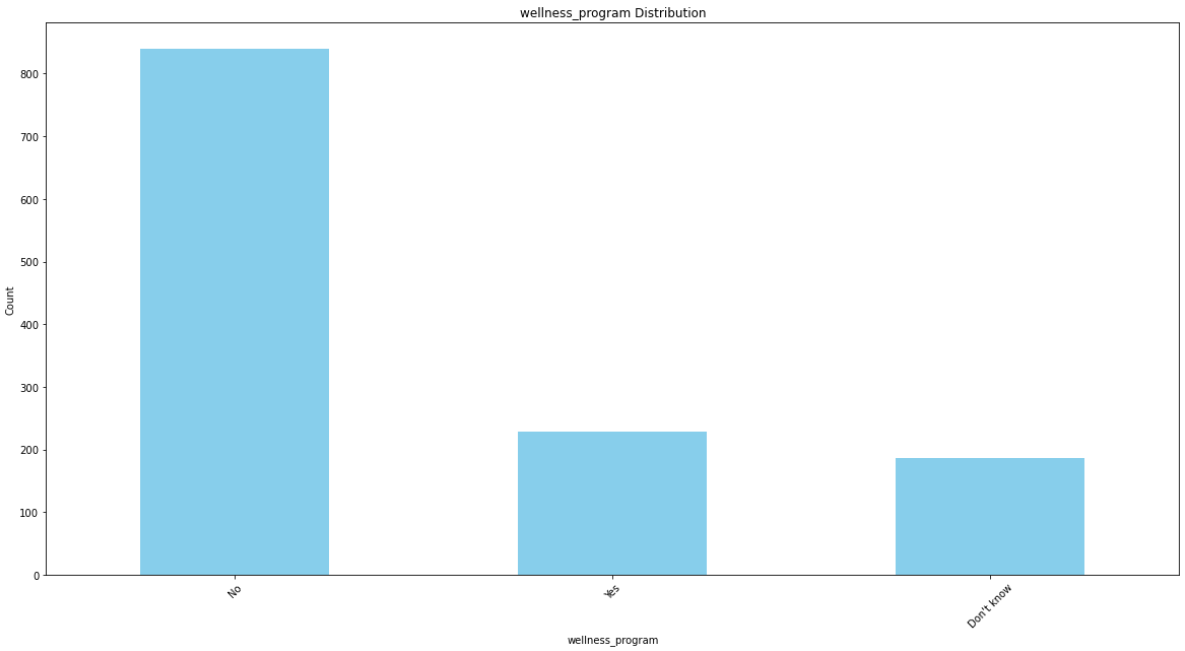

no_employees Distribution
Count
300
250
200
150
100
50
0
6-25
26-100
More than 1000
100-500
1-5
500-1000
no_employees

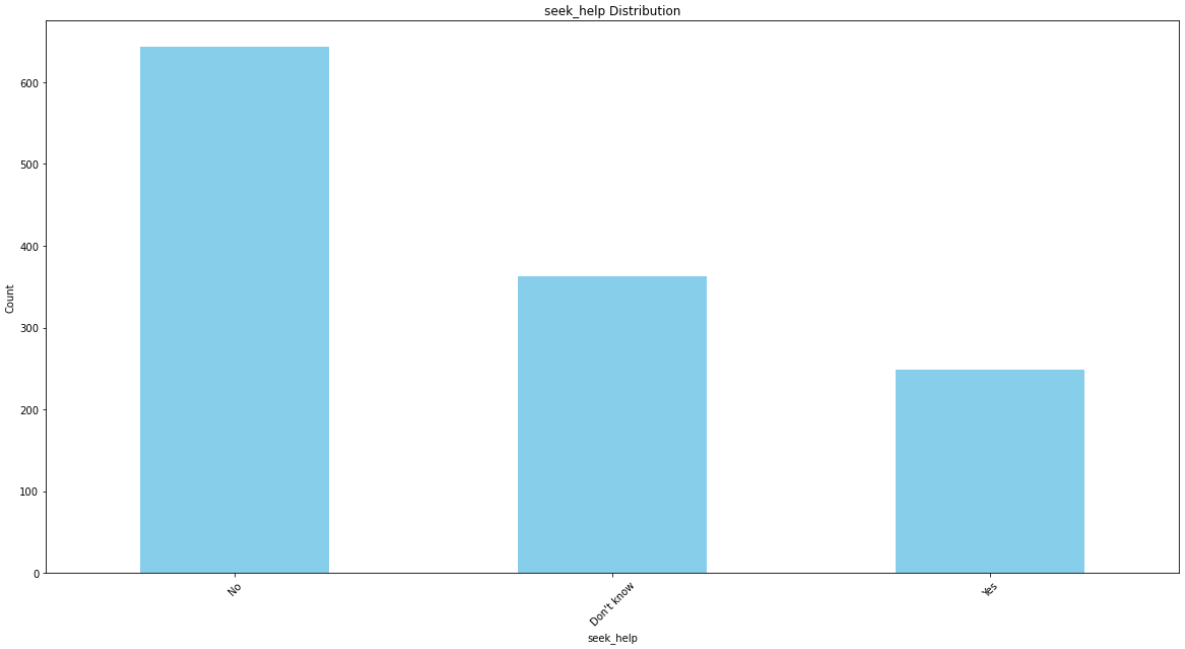

remote_work Distribution
Count
800
600
400
200
0
No
Yes
remote_work

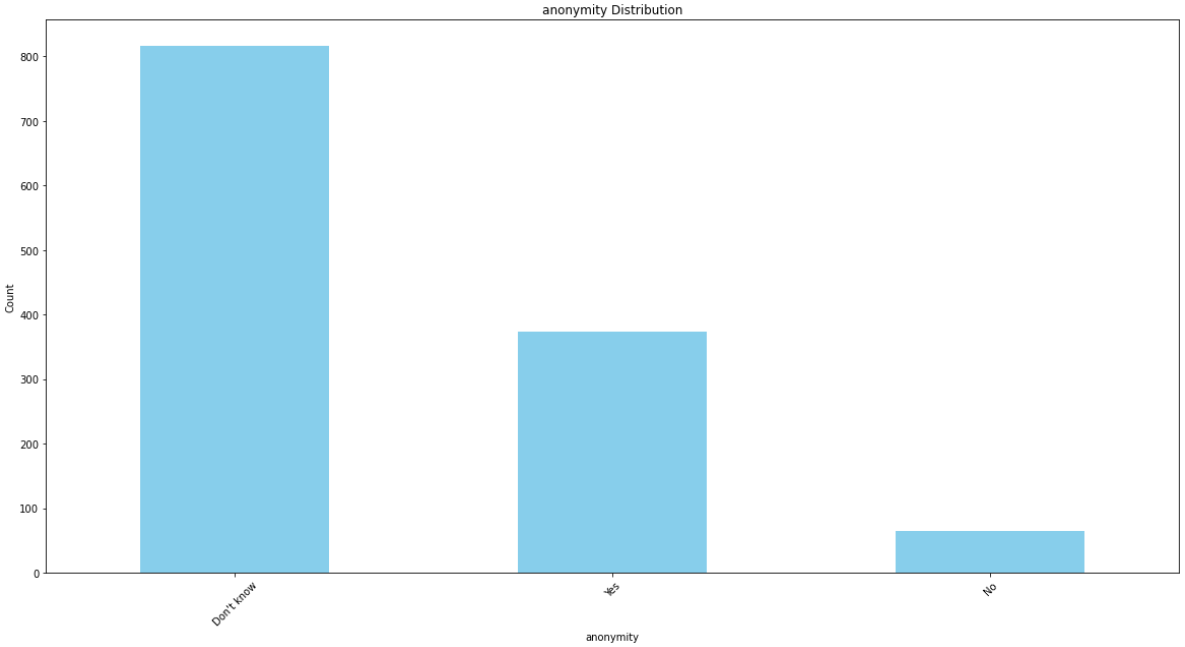

tech_company Distribution
Count
1000
800
600
400
200
0
Yes
No
tech_company

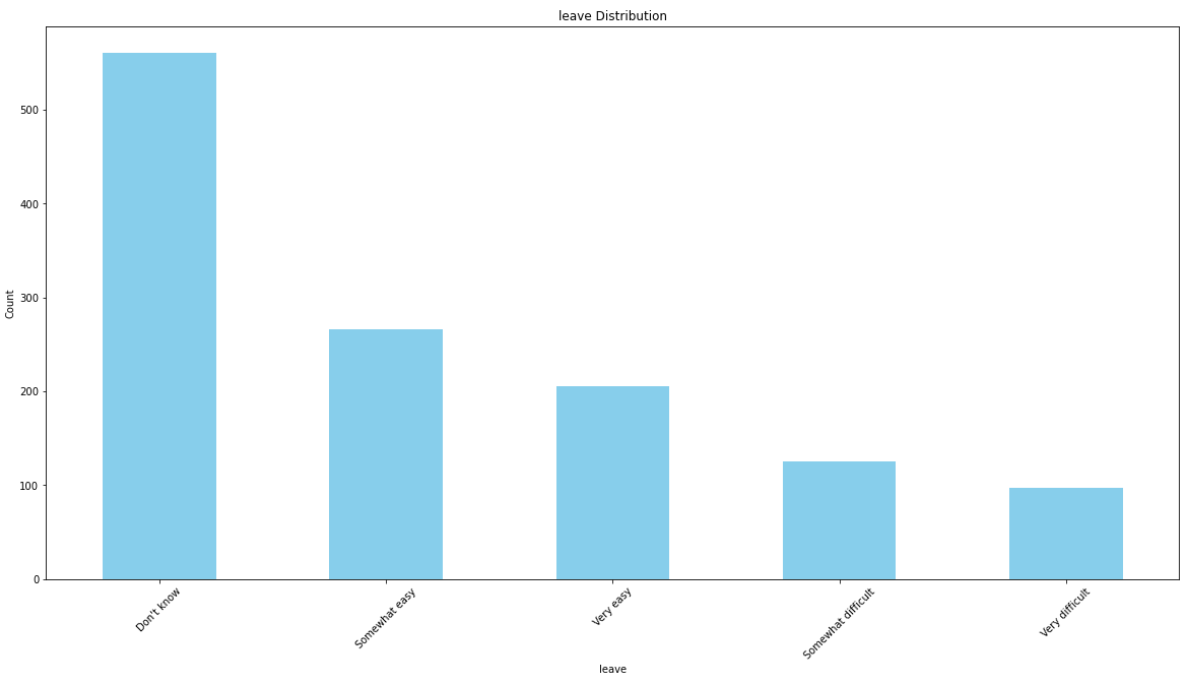

benefits Distribution
Count
0
100
200
300
400
Yes
Don't know
No
benefits

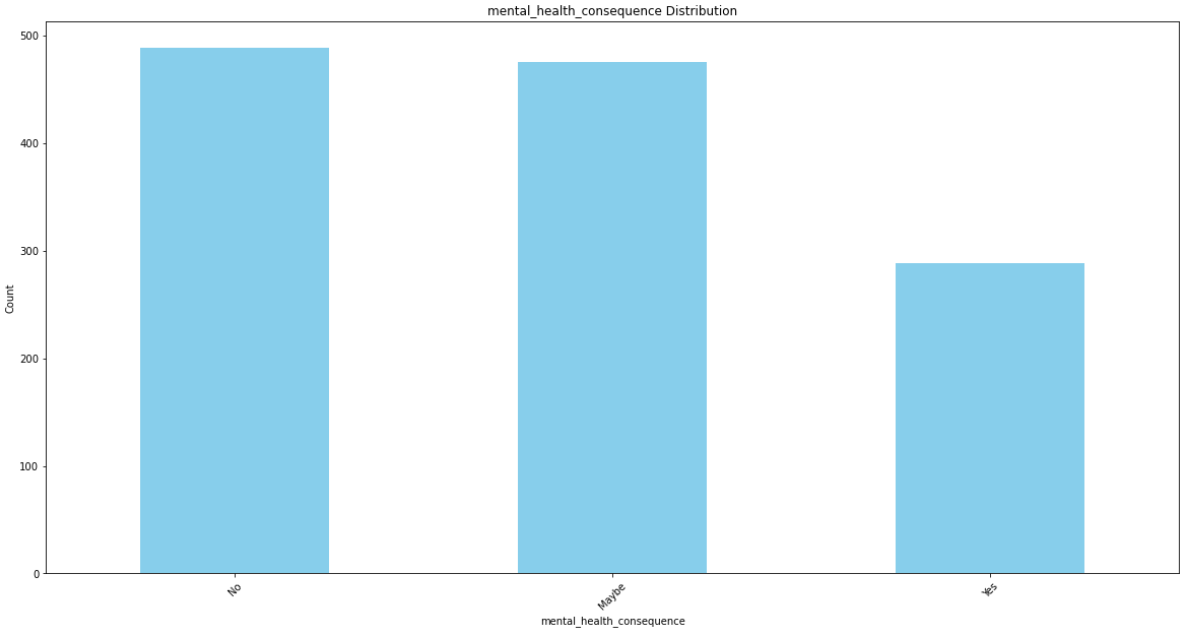

care_options Distribution
Count
0
100
200
300
400
500
No
Yes
Not sure
care_options

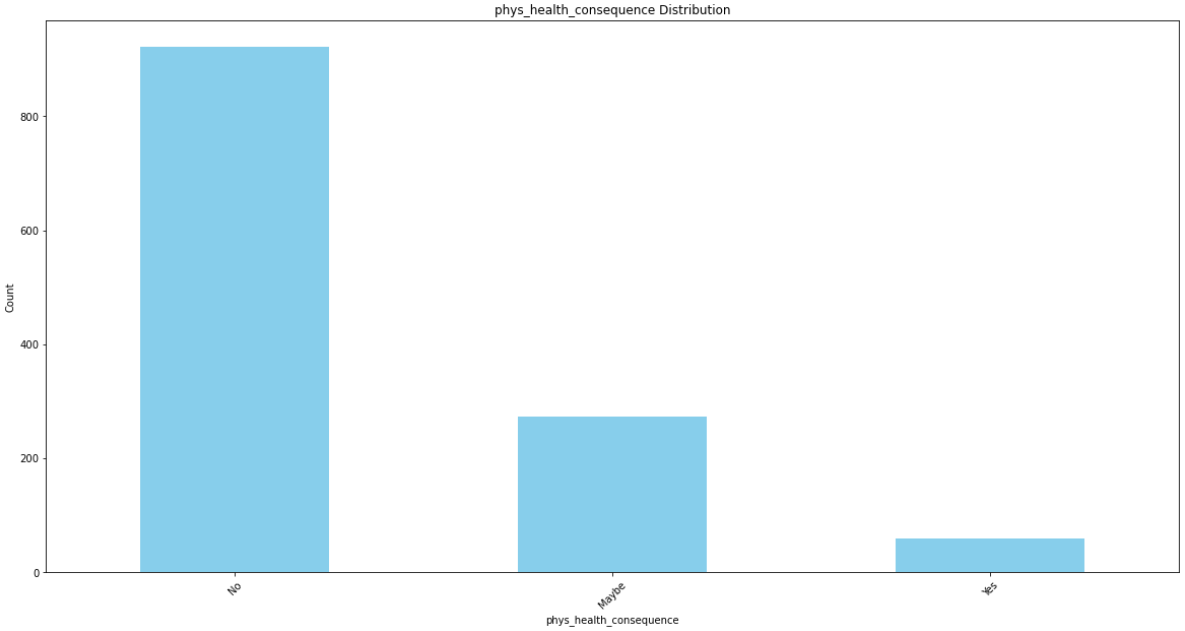

wellness_program Distribution
Count
0
100
200
300
400
500
600
700
800
No
Yes
Don't know
wellness_program

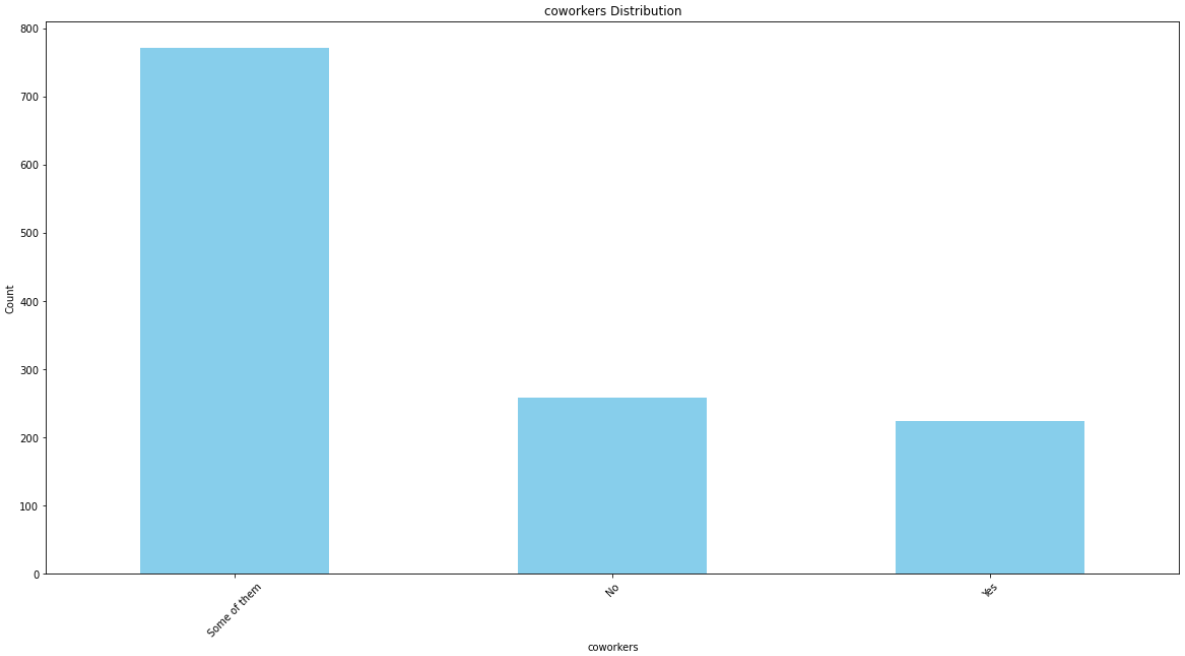

seek_help Distribution
Count
0
100
200
300
400
500
600
No
Don't know
Yes
seek_help

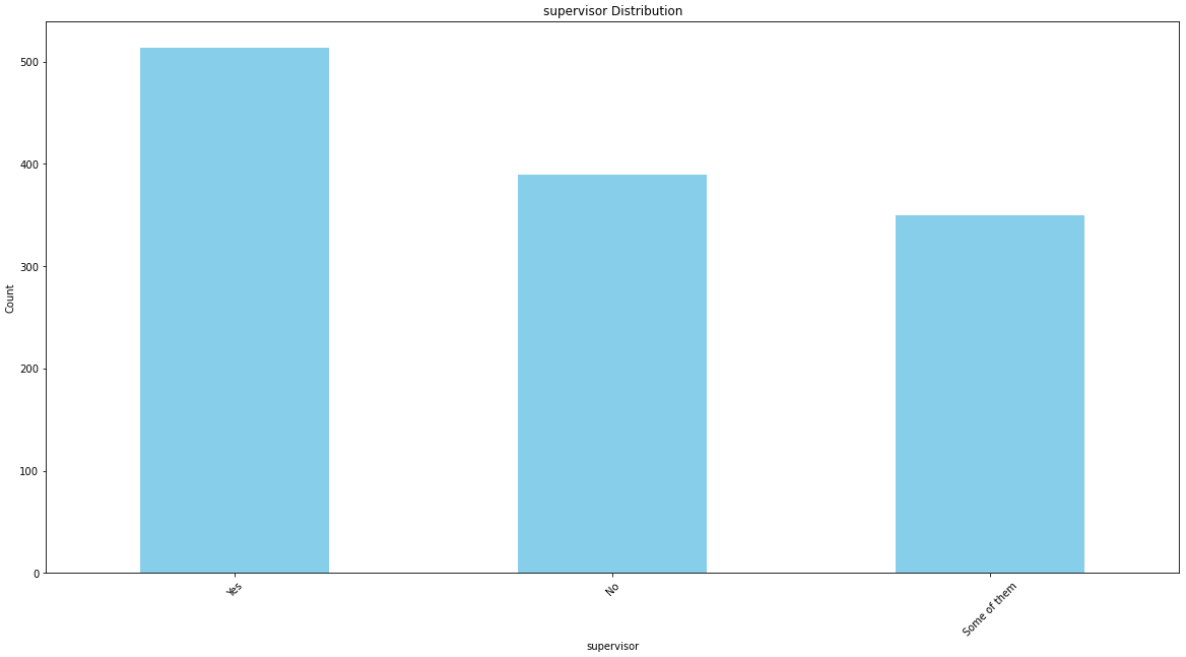

anonymity Distribution
Count
0
100
200
300
400
500
600
700
800
Don't know
Yes
No
anonymity

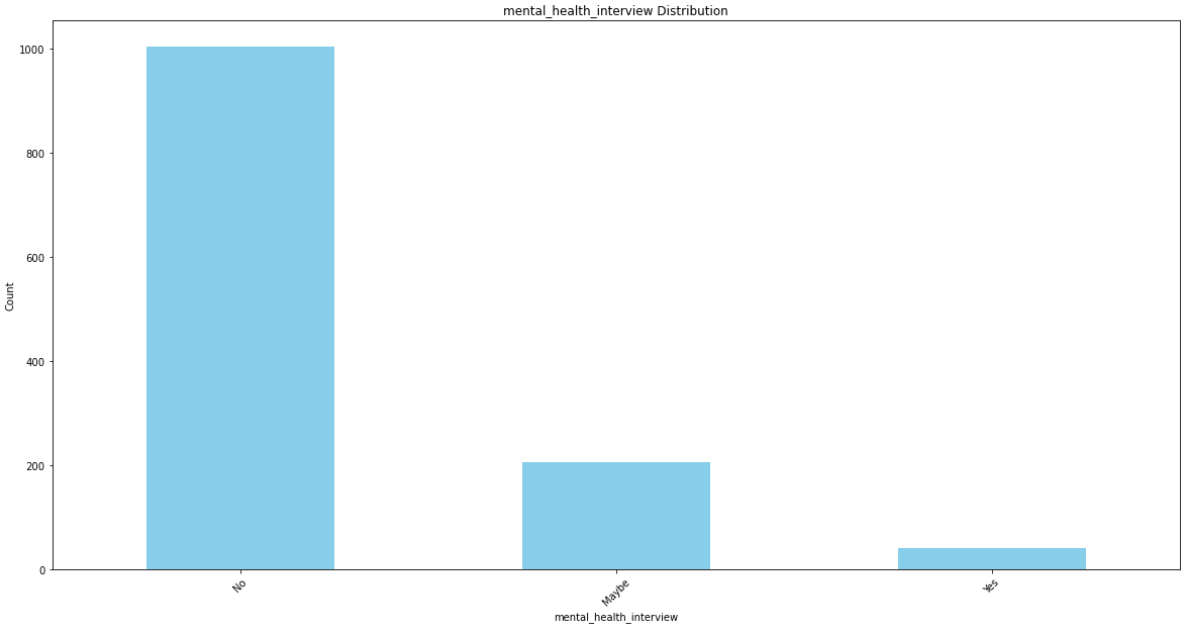

leave Distribution
Count
0
100
200
300
400
500
Don't know
Somewhat easy
Very easy
Somewhat difficult
Very difficult
leave

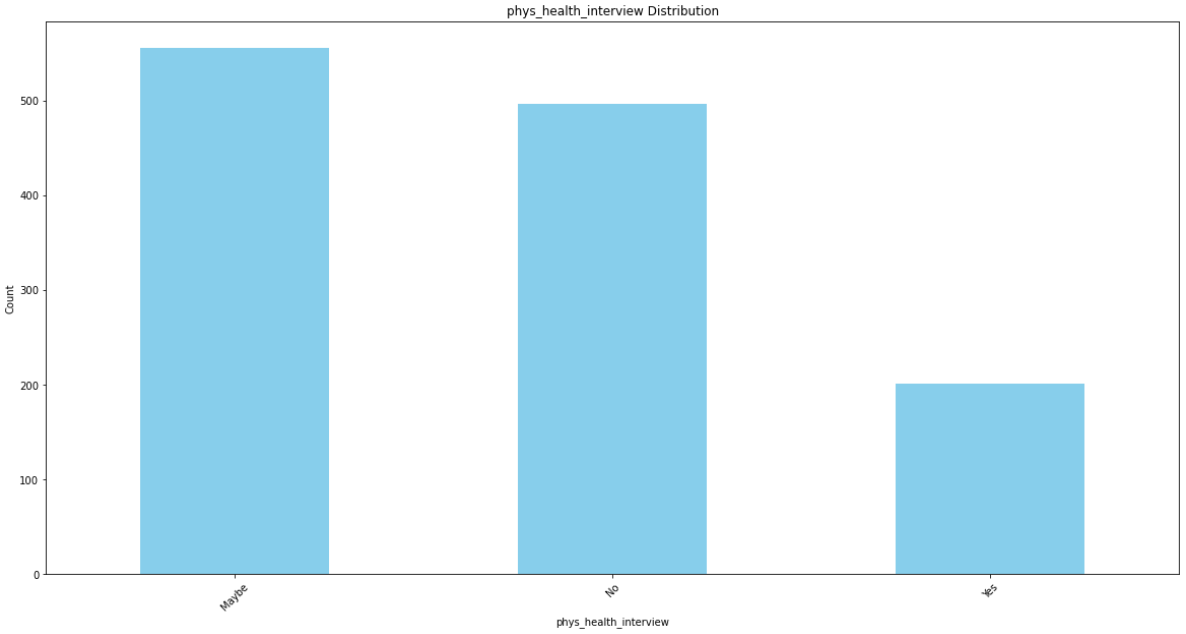

mental_health_consequence Distribution
Count
0
100
200
300
400
500
No
Maybe
Yes
mental_health_consequence

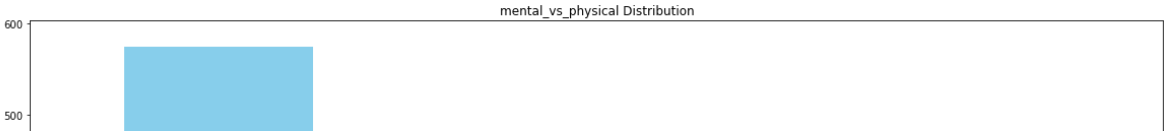

phys_health_consequence Distribution
Count
0
200
400
600
800
No
Maybe
Yes
phys_health_consequence


coworkers Distribution
Count
0
100
200
300
400
500
600
700
800
Some of them
No
Yes
coworkers

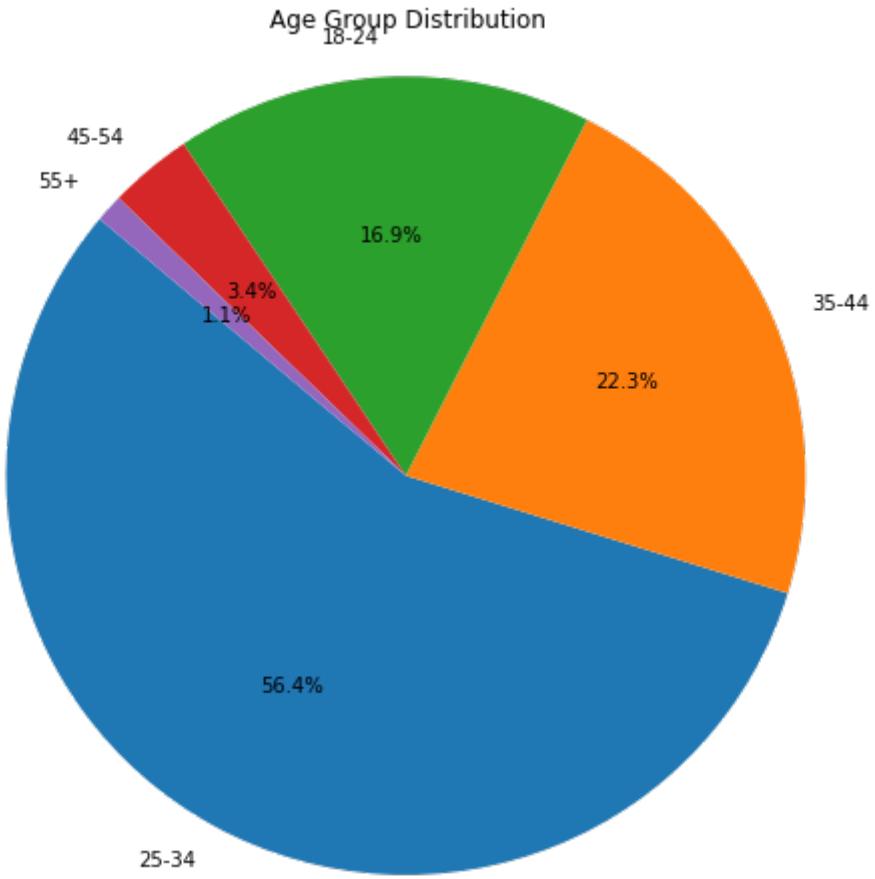

supervisor Distribution
Count
0
100
200
300
400
500
Yes
No
Some of them
supervisor


mental_health_interview Distribution
Count
0
200
400
600
800
1000
No
Maybe
Yes
mental_health_interview


phys_health_interview Distribution
Count
0
100
200
300
400
500
Maybe
No
Yes
phys_health_interview

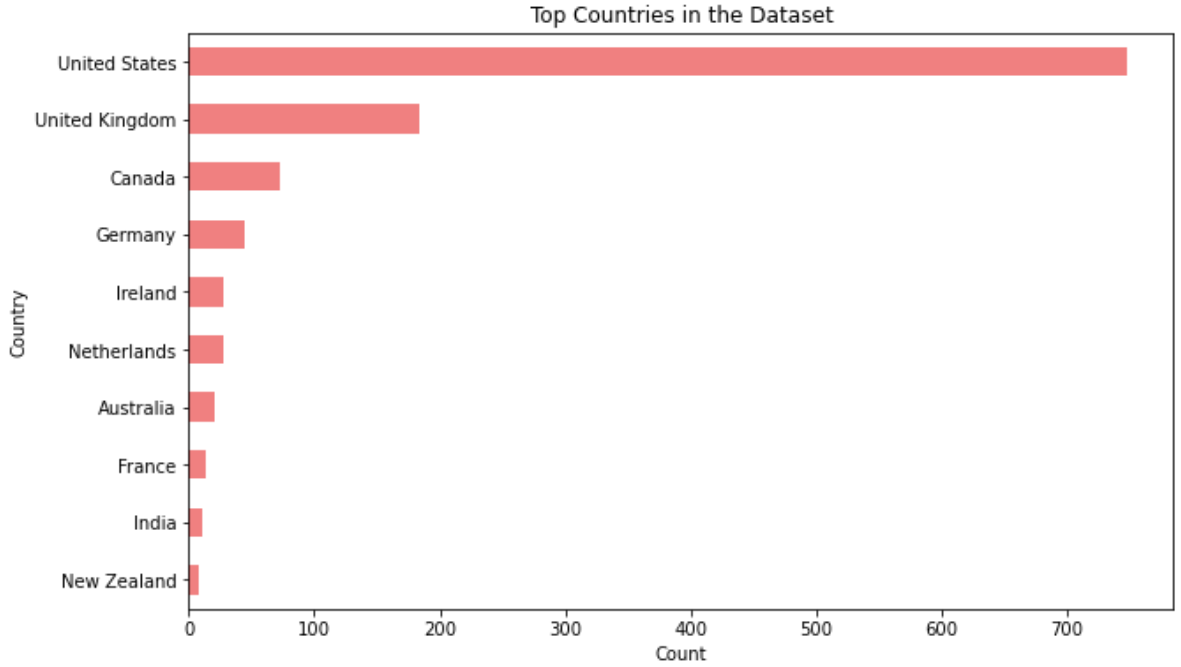


In [41]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Define age groups
bins = [18, 25, 35, 45, 55, 120]
labels = ['18-24', '25-34', '35-44', '45-54', '55+']

# Create a new column 'AgeGroup' based on the age categories
df['AgeGroup'] = pd.cut(df['Age'], bins=bins, labels=labels)

# Calculate the distribution of age groups
age_group_counts = df['AgeGroup'].value_counts()

# Create a pie chart
plt.figure(figsize=(8, 8))
plt.pie(age_group_counts, labels=age_group_counts.index, autopct='%1.1f%%'
plt.title('Age Group Distribution')
plt.axis('equal')  # Equal aspect ratio ensures that the pie chart is circ
plt.show()
```

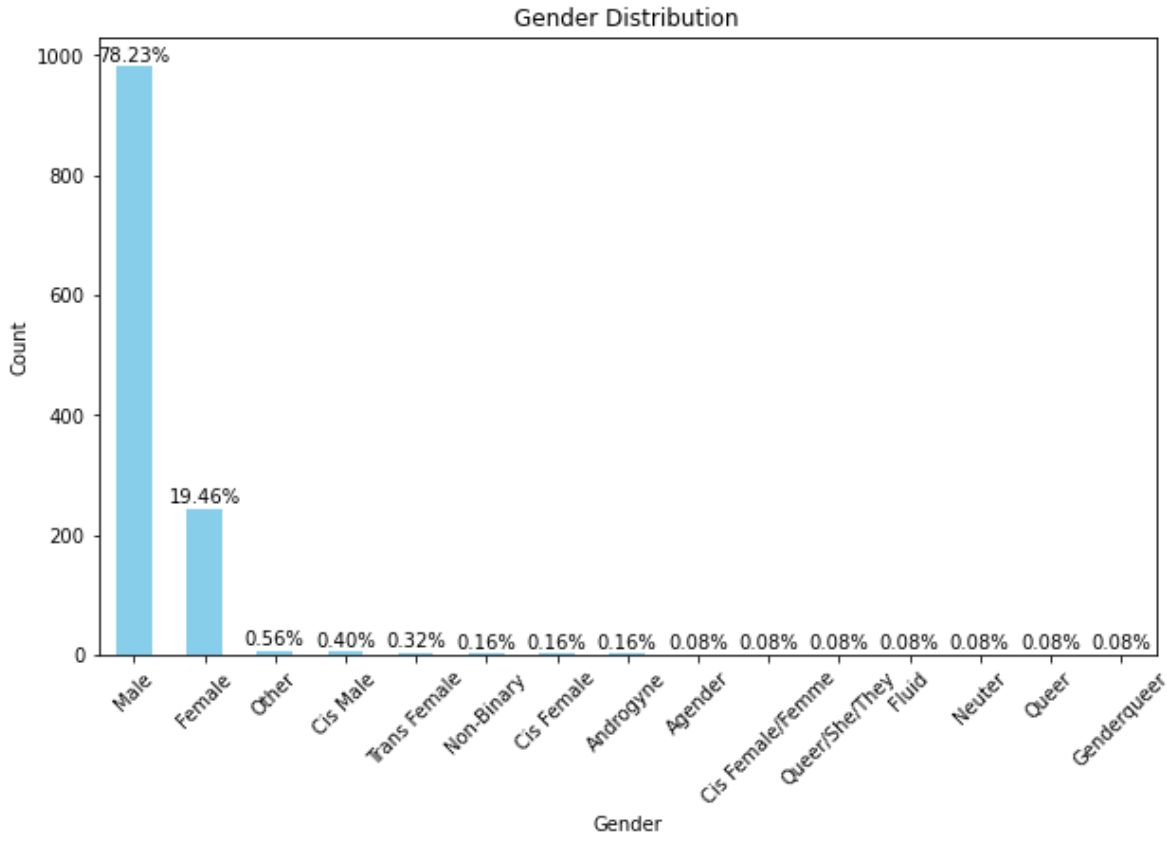


In [42]:

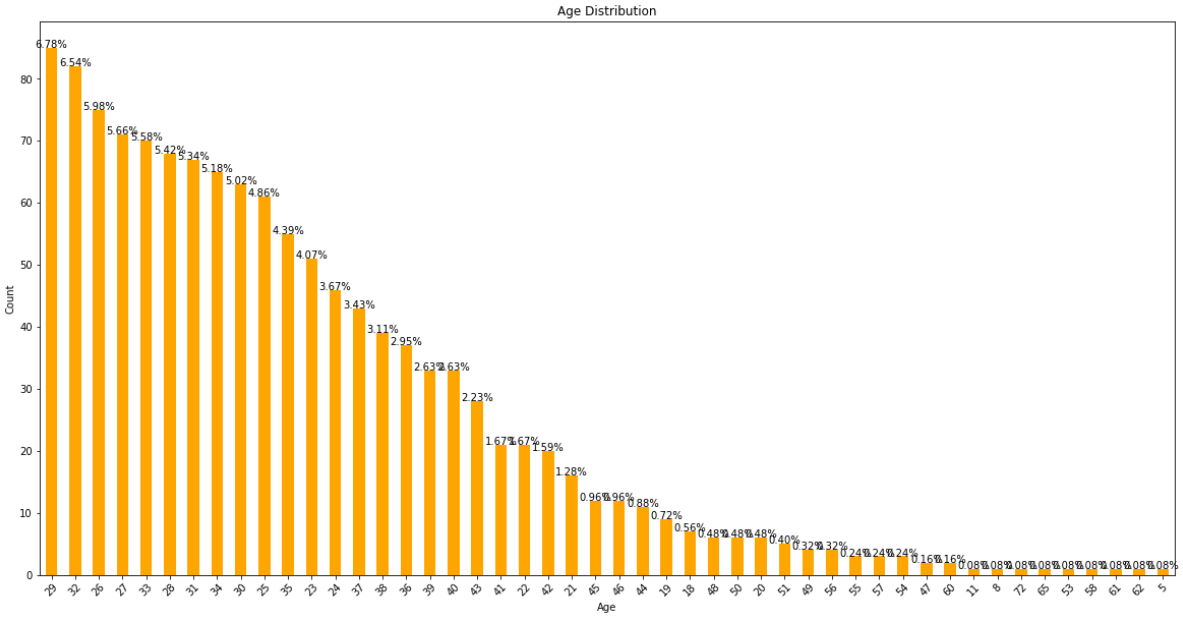
```
import pandas as pd
import matplotlib.pyplot as plt

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')


# Create horizontal bar chart for Country
country_counts = df['Country'].value_counts()
top_n = 10  # Choose how many countries to display
top_countries = country_counts.head(top_n)
plt.figure(figsize=(10, 6))
top_countries.plot(kind='barh', color='lightcoral')
plt.xlabel('Count')
plt.ylabel('Country')
plt.title('Top Countries in the Dataset')
plt.gca().invert_yaxis()
plt.show()
```

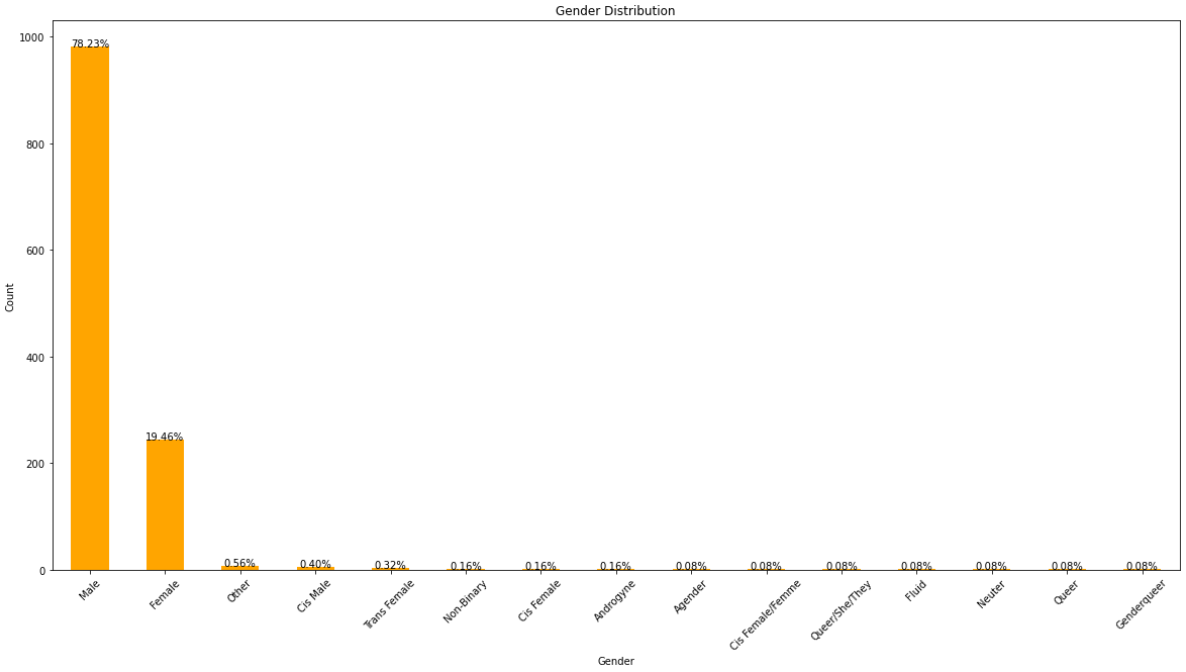


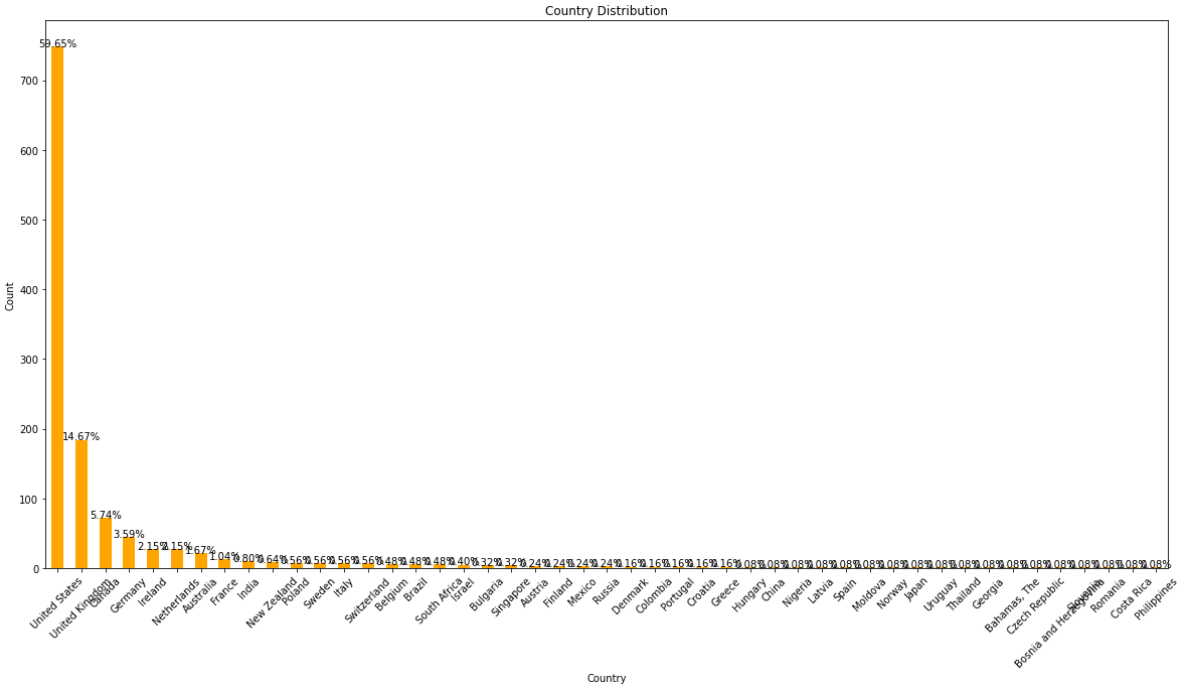
In [43]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Create a bar chart for Gender distribution with percentage labels
gender_counts = df['Gender'].value_counts()
total_count = len(df)
percentage_values = (gender_counts / total_count) * 100

plt.figure(figsize=(10, 6))
bars = gender_counts.plot(kind='bar', color='skyblue')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.title('Gender Distribution')
plt.xticks(rotation=45)

# Add percentage labels on top of each bar
for i, count in enumerate(gender_counts):
    plt.text(i, count + 10, f'{percentage_values[i]:.2f}%', ha='center')

plt.show()
```

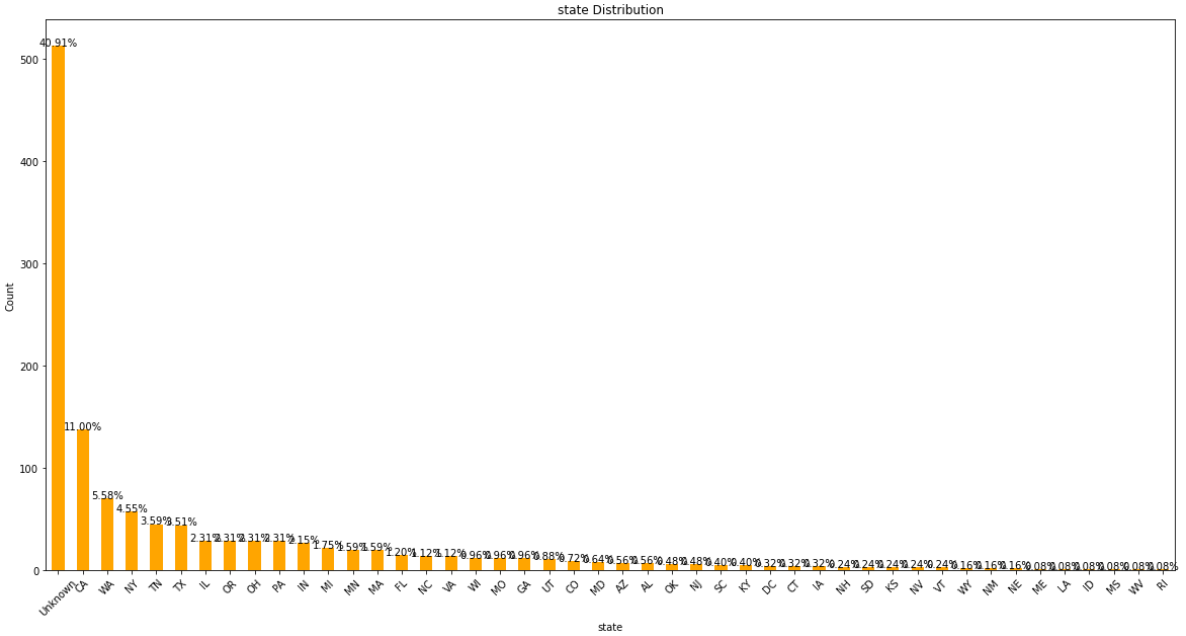


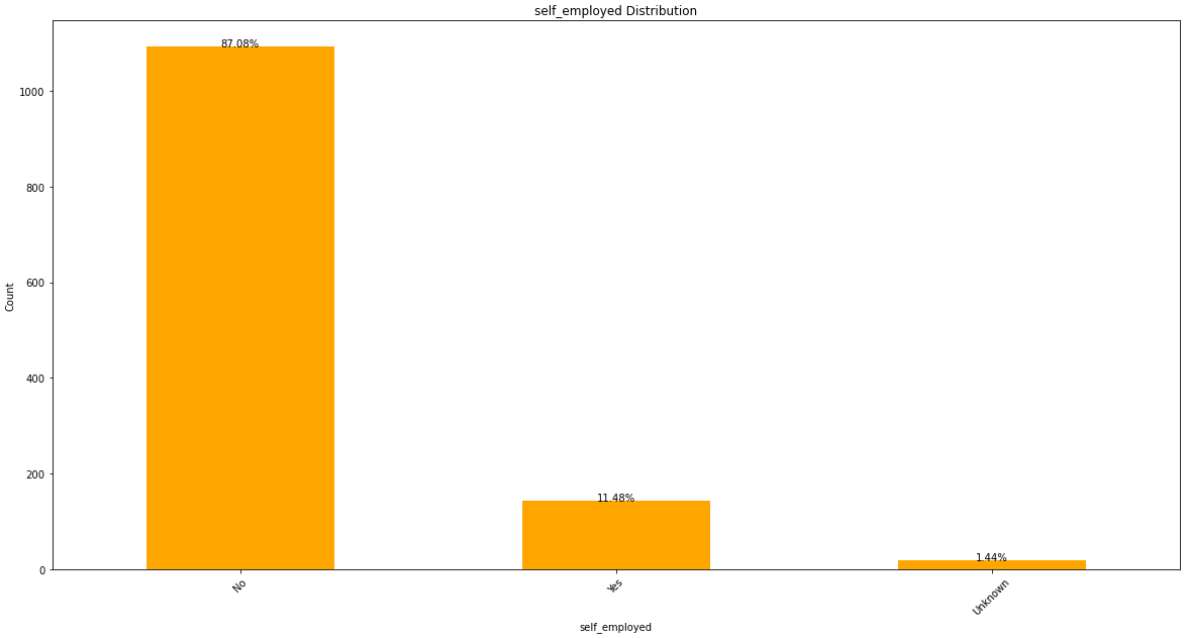
In [48]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Read the dataset
df = pd.read_csv('cleaned_survey.csv')

# Define a list of columns to visualize with percentages
columns_to_visualize = [
    'Age', 'Gender', 'Country', 'state', 'self_employed', 'family_history'
    'treatment', 'work_interfere', 'no_employees', 'remote_work',
    'tech_company', 'benefits', 'care_options', 'wellness_program', 'seek_
    'anonymity', 'leave', 'mental_health_consequence', 'phys_health_conseq
    'coworkers', 'supervisor', 'mental_health_interview', 'phys_health_int
    'mental_vs_physical', 'obs_consequence'
]

# Customize visualizations for each column
for column in columns_to_visualize:
    if column in ['Age', 'Gender', 'Country','state', 'self_employed', 'fa
    'treatment', 'work_interfere', 'no_employees', 'remote_work',
    'tech_company', 'benefits', 'care_options', 'wellness_program', 'seek_
    'anonymity', 'leave', 'mental_health_consequence', 'phys_health_conseq
    'coworkers', 'supervisor', 'mental_health_interview', 'phys_health_int
    'mental_vs_physical', 'obs_consequence']:
        # Create a bar chart for selected columns
        column_counts = df[column].value_counts()
        plt.figure(figsize=(20, 10))
        ax = column_counts.plot(kind='bar', color='orange')
        plt.xlabel(column)
        plt.ylabel('Count')
        plt.title(f'{column} Distribution')
        plt.xticks(rotation=45)

        # Add percentage labels to the bars
        total_count = len(df[column])
        for p in ax.patches:
            percentage = f"{100 * p.get_height() / total_count:.2f}%"
            ax.annotate(percentage, (p.get_x() + p.get_width() / 2, p.get_

        plt.show()
```

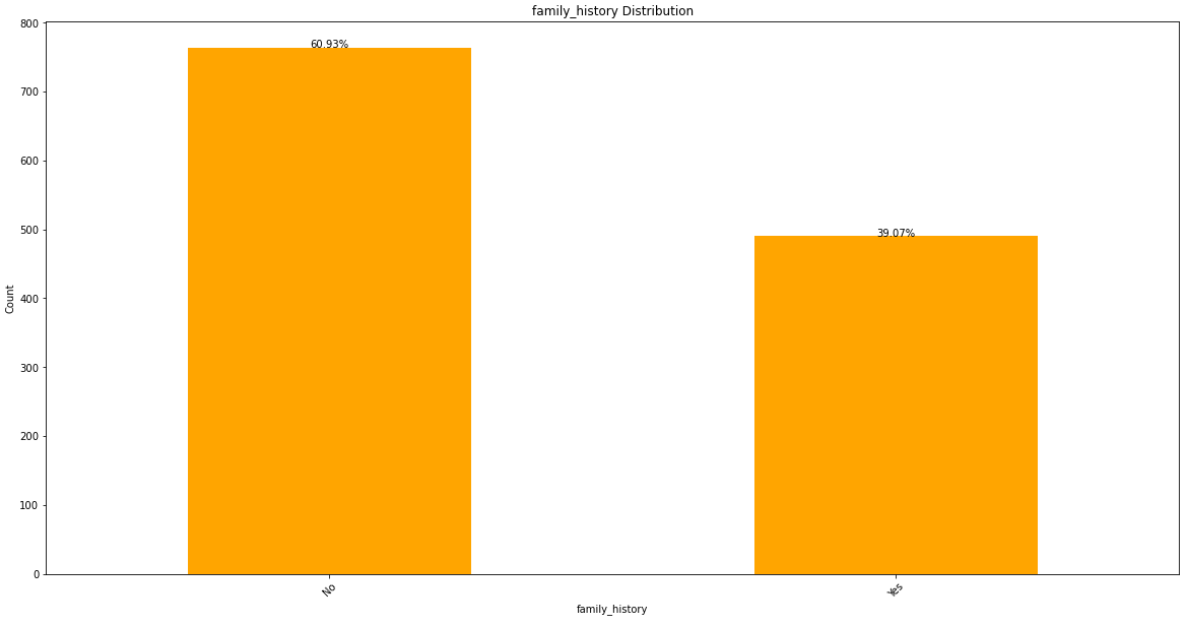


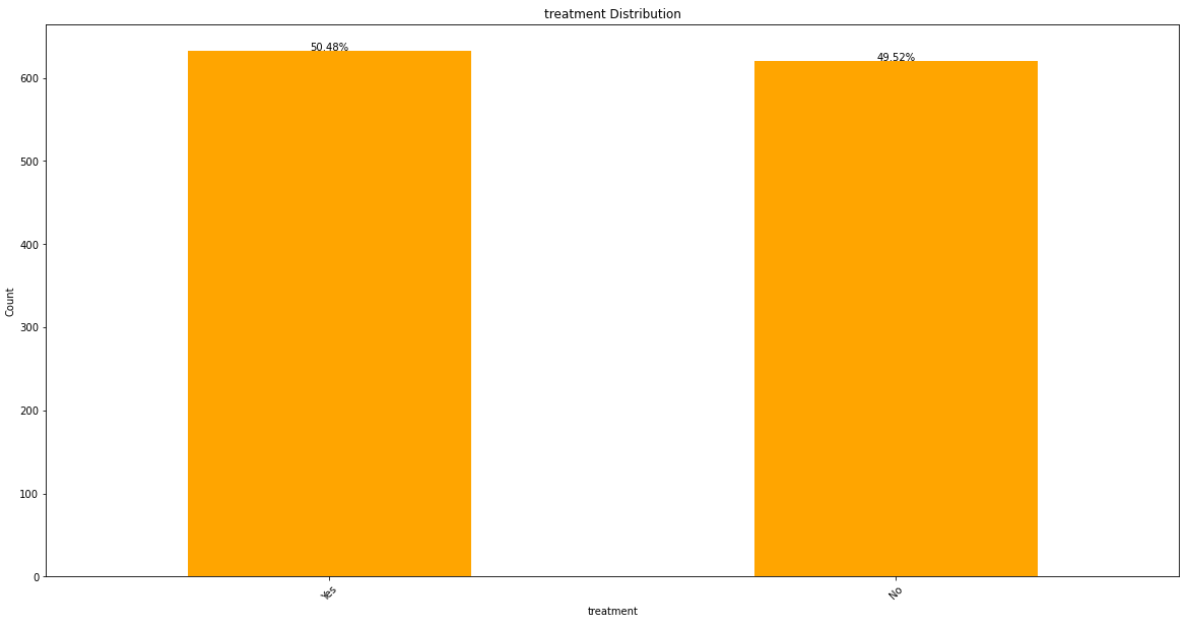

Gender Distribution
Count
Gender
Male
Female
Other
Cis Male
Trans Female
Non-Binary
Cis Female
Androgyne
Agender
Cis Female/Femme
Queer/She/They
Fluid
Neuter
Queer
Genderqueer
78.23%
19.46%
0.56%
0.40%
0.32%
0.16%
0.16%
0.16%
0.08%
0.08%
0.08%
0.08%
0.08%
0.08%
0.08%

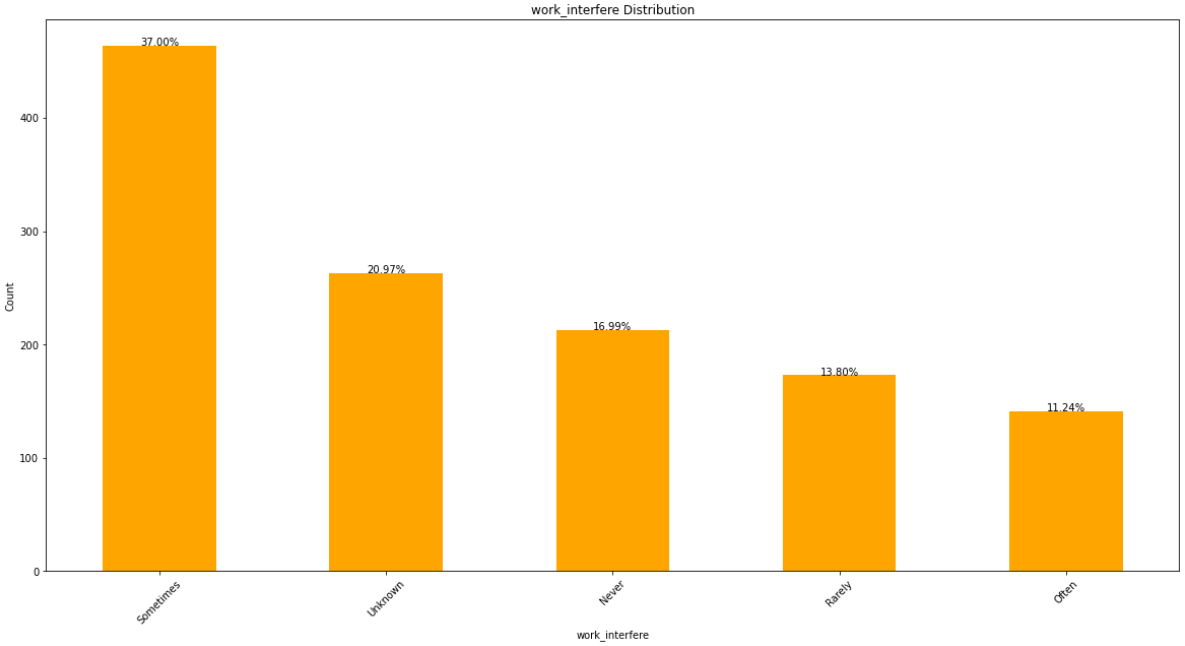

Country Distribution
Count
Country
United States
United Kingdom
Canada
Germany
Ireland
Netherlands
Australia
France
India
New Zealand
Poland
Sweden
Italy
Switzerland
Belgium
Brazil
South Africa
Israel
Bulgaria
Singapore
Austria
Finland
Mexico
Russia
Denmark
Colombia
Portugal
Croatia
Greece
Hungary
China
Nigeria
Latvia
Spain
Moldova
Norway
Japan
Uruguay
Thailand
Georgia
Bahamas, The
Czech Republic
Bosnia and Herzegovina
Slovenia
Romania
Costa Rica
Philippines
59.65%
14.67%
5.74%
3.59%

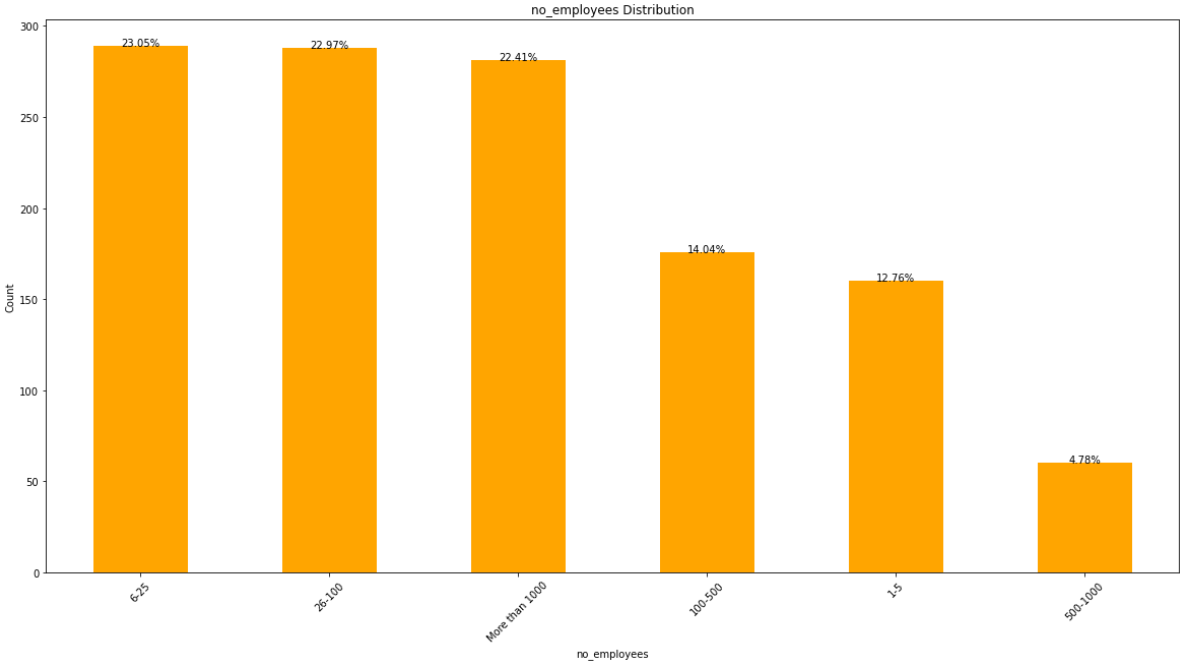

state Distribution
Count
state
Unknown
CA
WA
NY
TN
TX
IL
OR
OH
PA
IN
MI
MN
MA
FL
NC
VA
WI
MO
GA
UT
CO
MD
AZ
AL
OK
NJ
SC
KY
DC
CT
IA
NH
SD
KS
NV
VT
WY
NM
NE
ME
LA
ID
MS
WV
RI
40.91%
11.00%
5.58%
4.55%
3.59%
3.51%

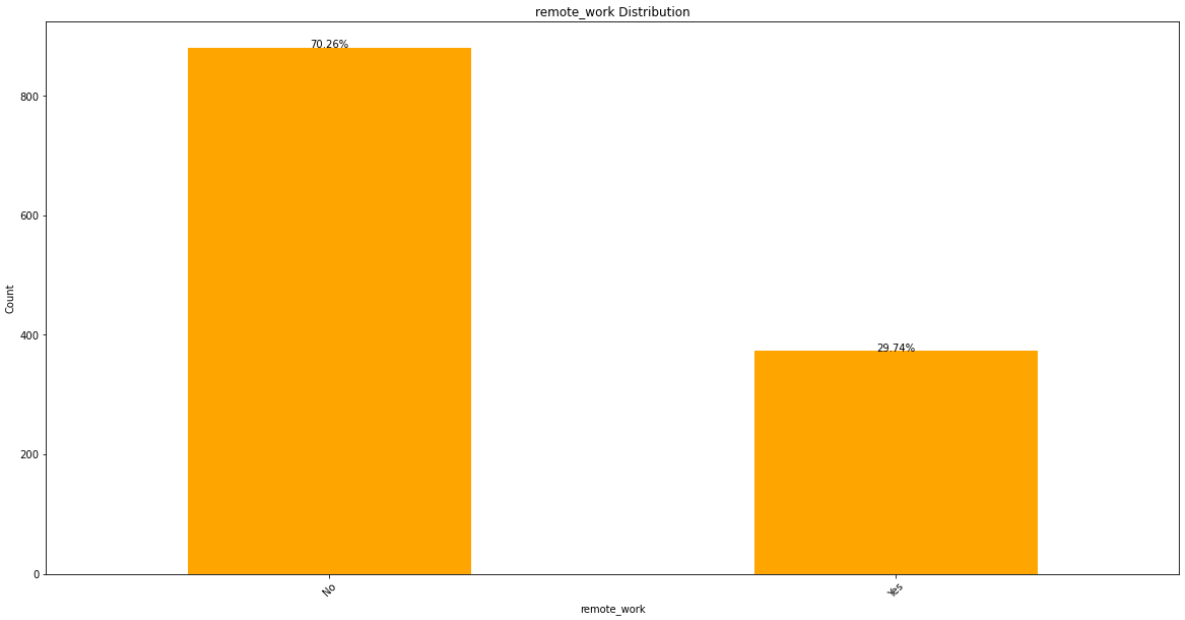

self_employed Distribution
87.08%
11.48%
1.44%
Count
0
200
400
600
800
1000
No
Yes
Unknown
self_employed

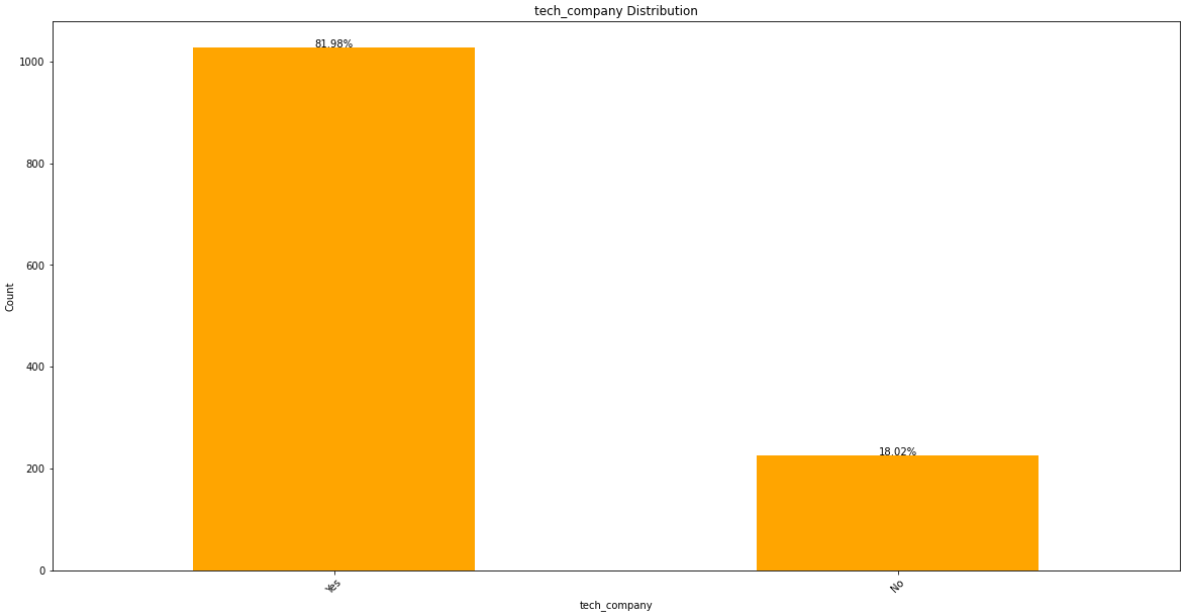

family_history Distribution
60.93%
39.07%
Count
0
100
200
300
400
500
600
700
800
No
Yes
family_history

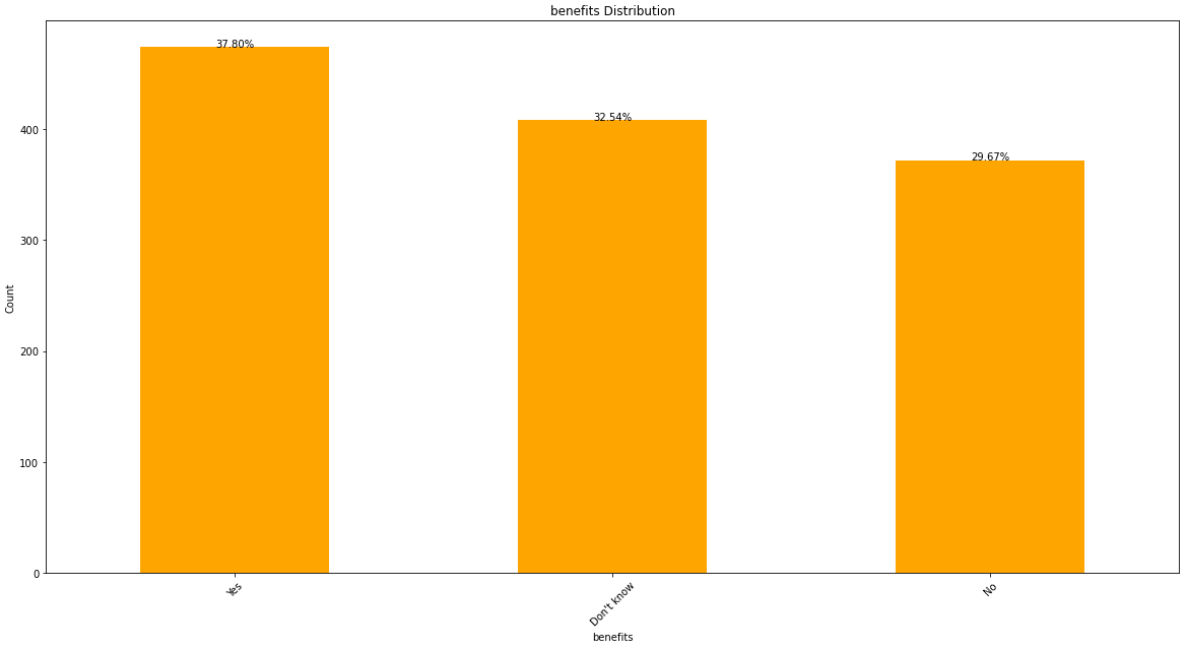

treatment Distribution
50.48%
49.52%
Count
0
100
200
300
400
500
600
Yes
No
treatment

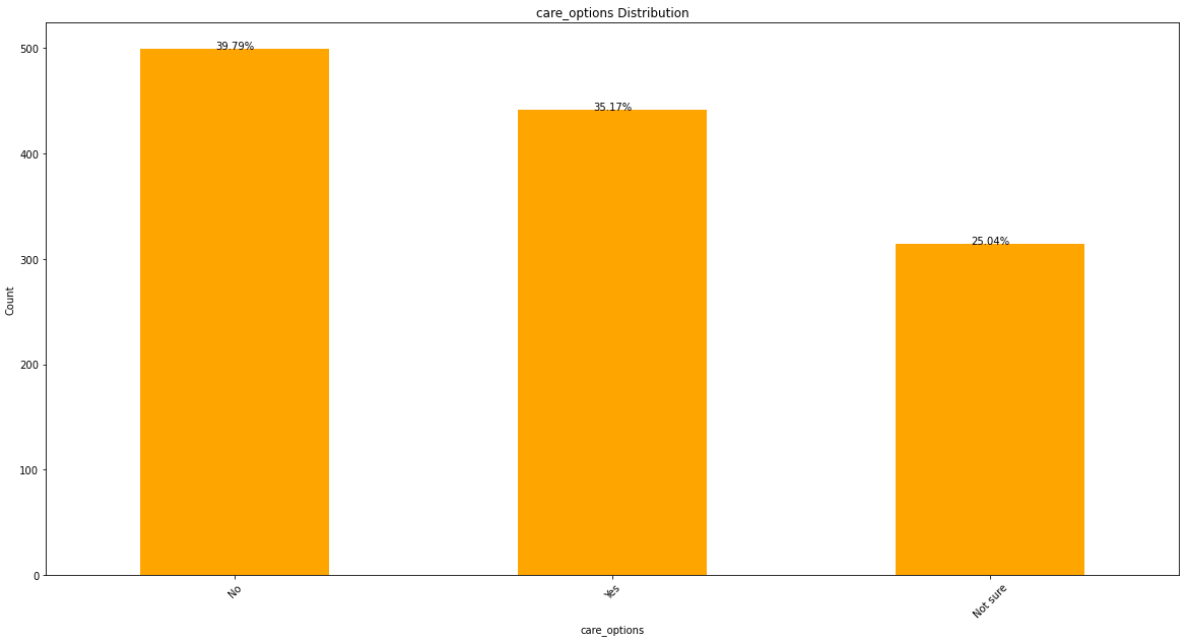

work_interfere Distribution
37.00%
20.97%
16.99%
13.80%
11.24%
Count
0
100
200
300
400
Sometimes
Unknown
Never
Rarely
Often
work_interfere

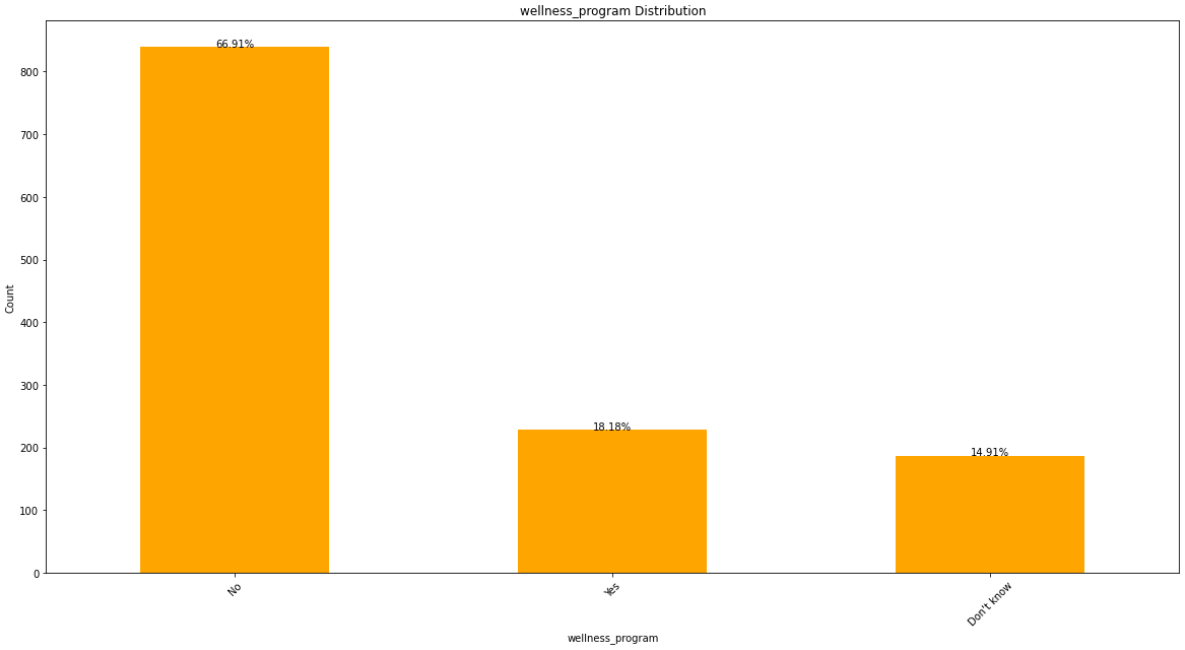

no_employees Distribution
23.05%
22.97%
22.41%
14.04%
12.76%
4.78%
Count
0
50
100
150
200
250
300
6-25
26-100
More than 1000
100-500
1-5
500-1000
no_employees

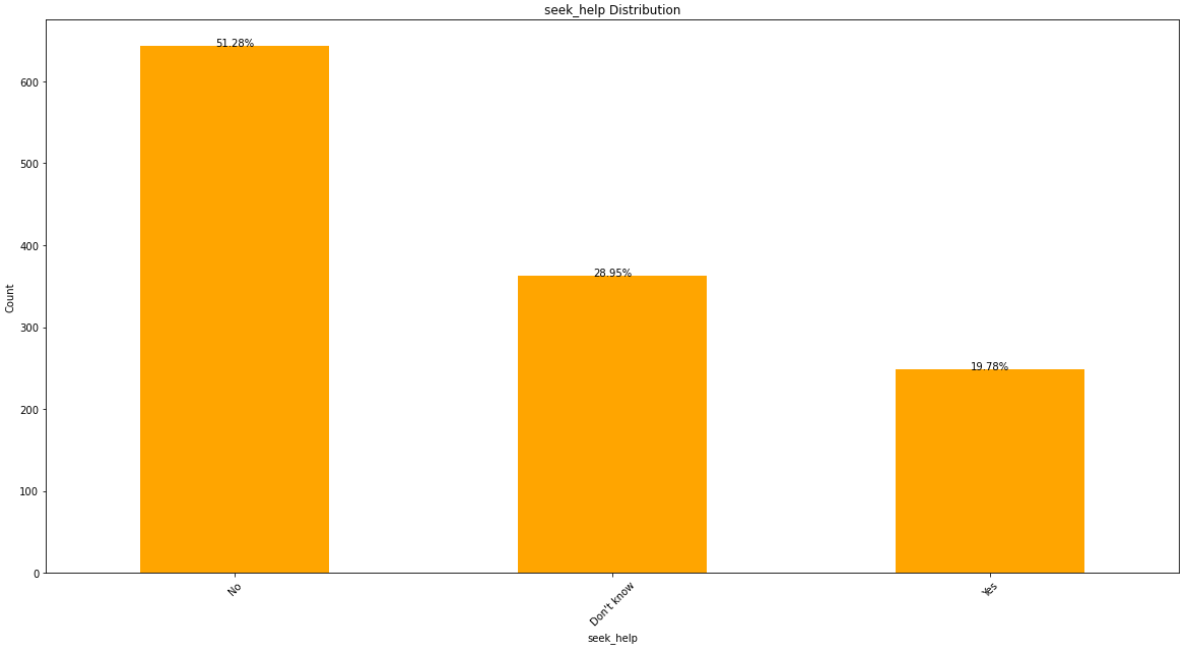

remote_work Distribution
70.26%
29.74%
Count
0
200
400
600
800
No
Yes
remote_work

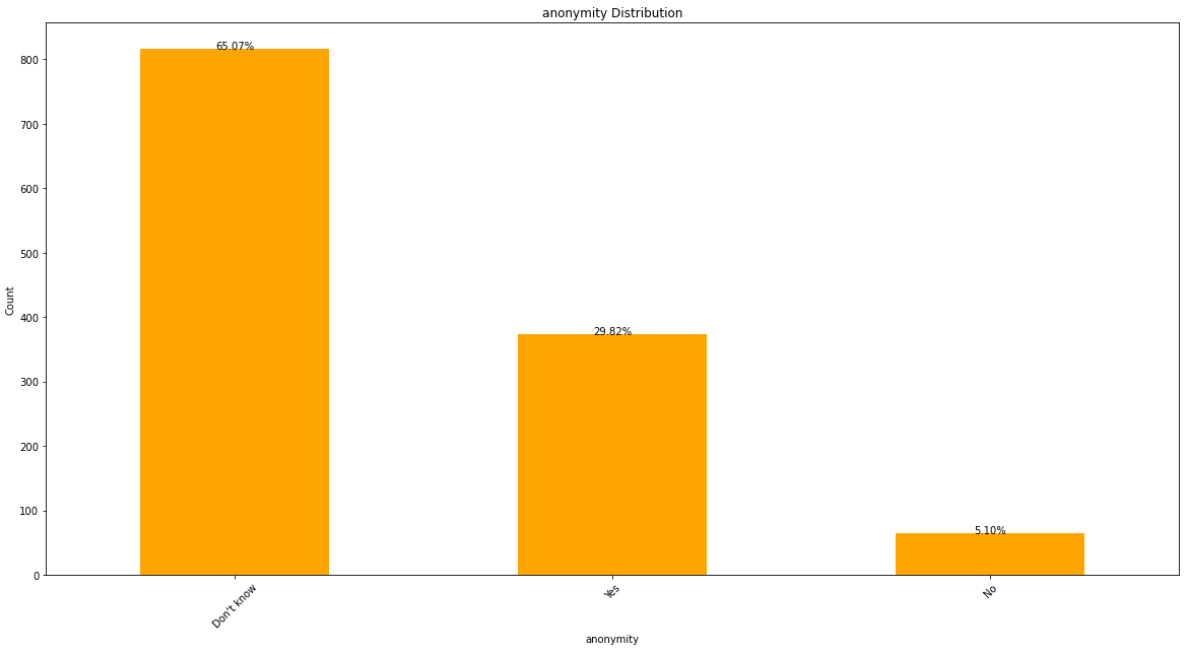

tech_company Distribution
81.98%
18.02%
Count
tech_company
Yes
No

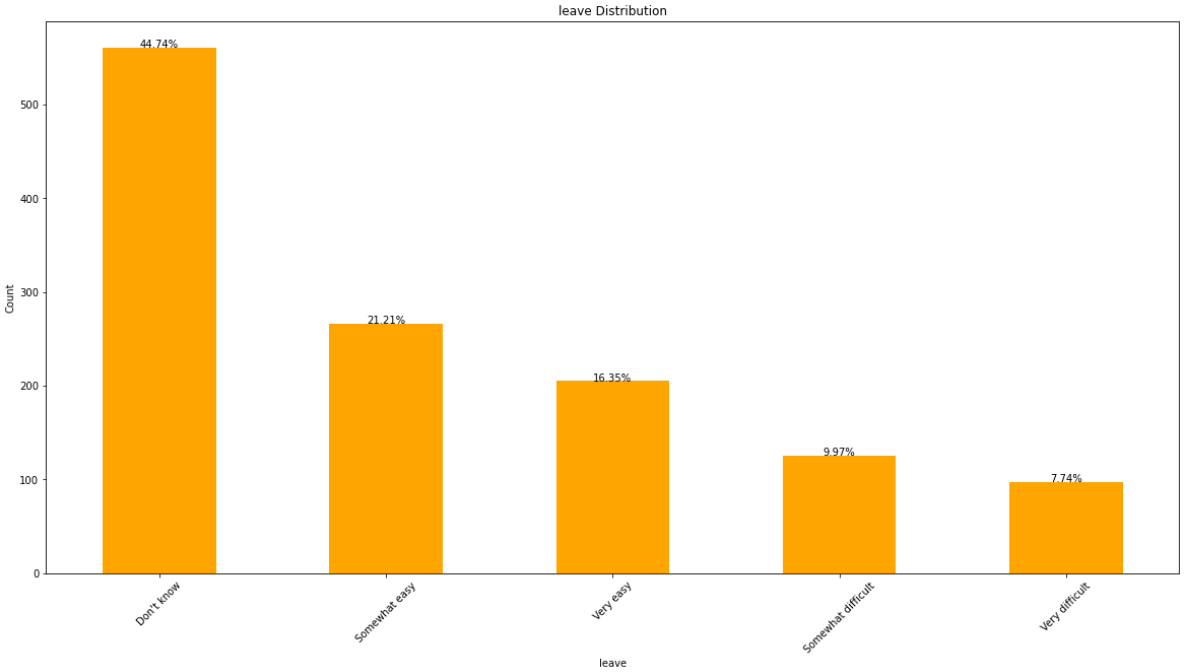

benefits Distribution
37.80%
32.54%
29.67%
Count
benefits
Yes
Don't know
No

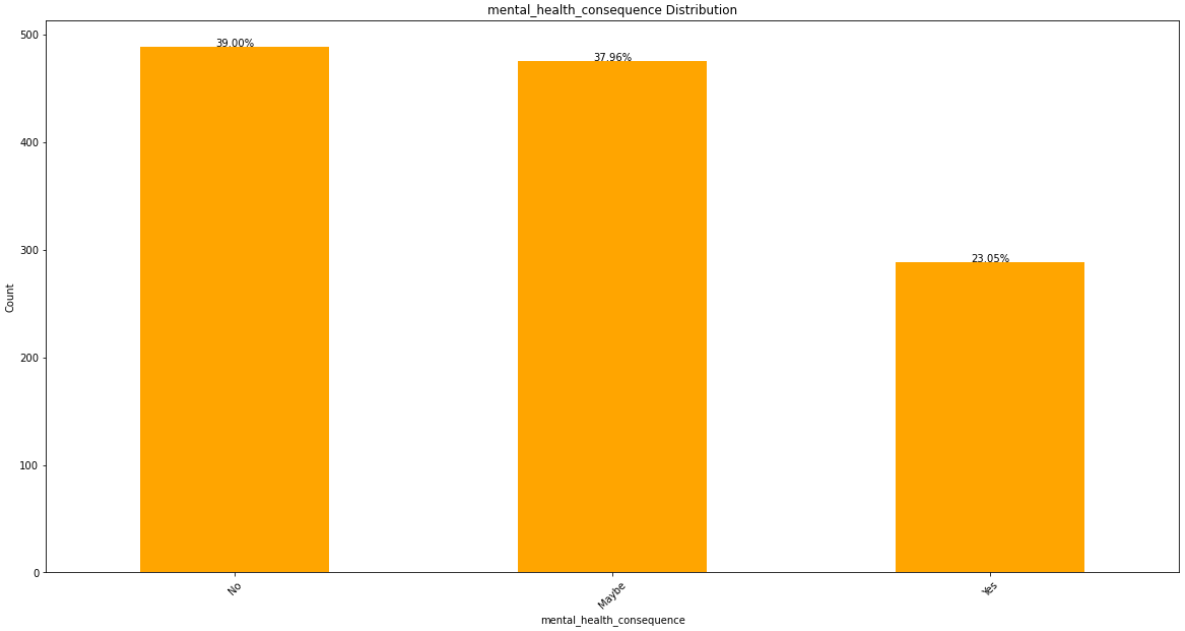

care_options Distribution
39.79%
35.17%
25.04%
Count
care_options
No
Yes
Not sure

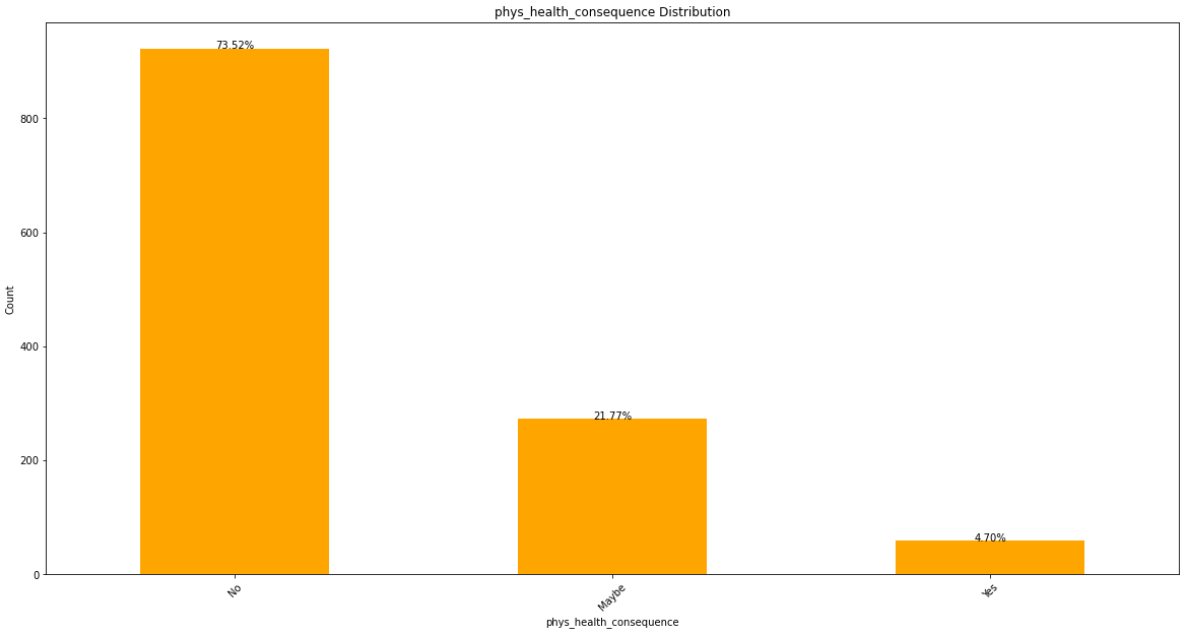

wellness_program Distribution
66.91%
18.18%
14.91%
Count
0
100
200
300
400
500
600
700
800
No
Yes
Don't know
wellness_program

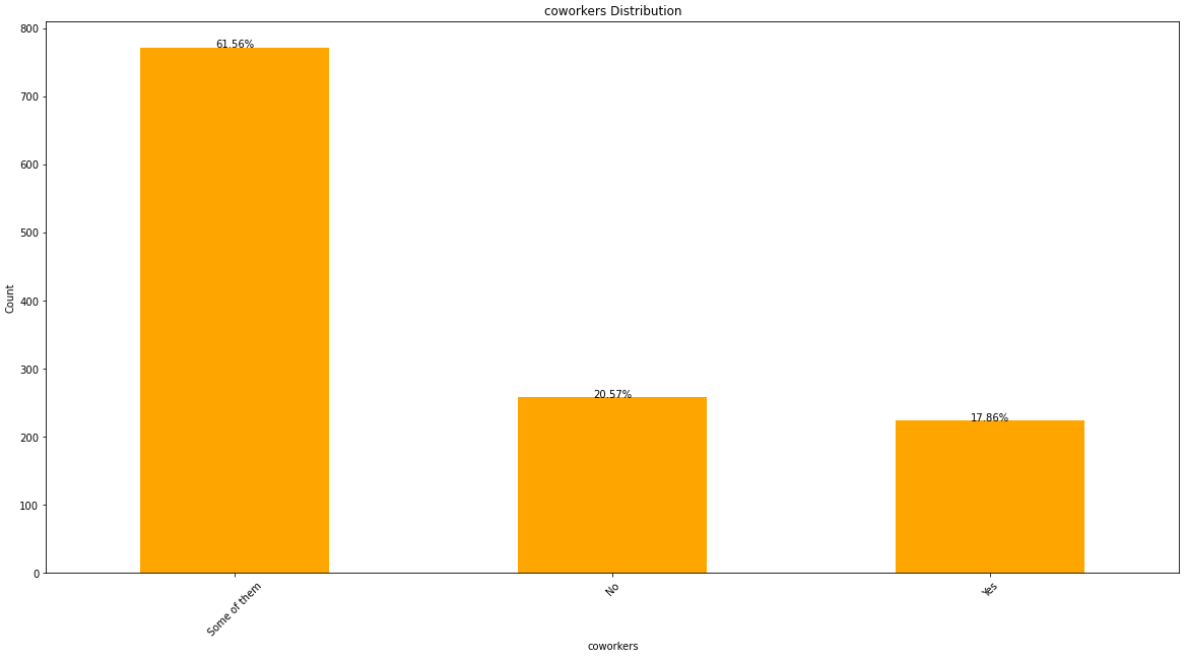

seek_help Distribution
51.28%
28.95%
19.78%
Count
0
100
200
300
400
500
600
No
Don't know
Yes
seek_help

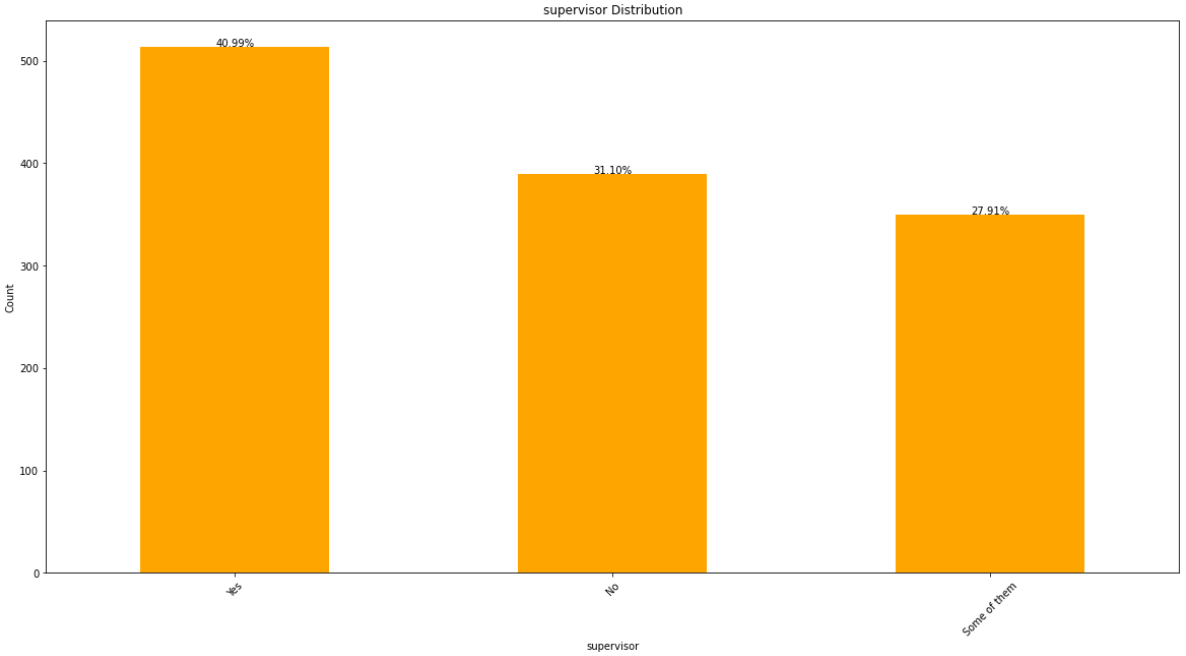

anonymity Distribution
65.07%
29.82%
5.10%
Count
0
100
200
300
400
500
600
700
800
Don't know
Yes
No
anonymity

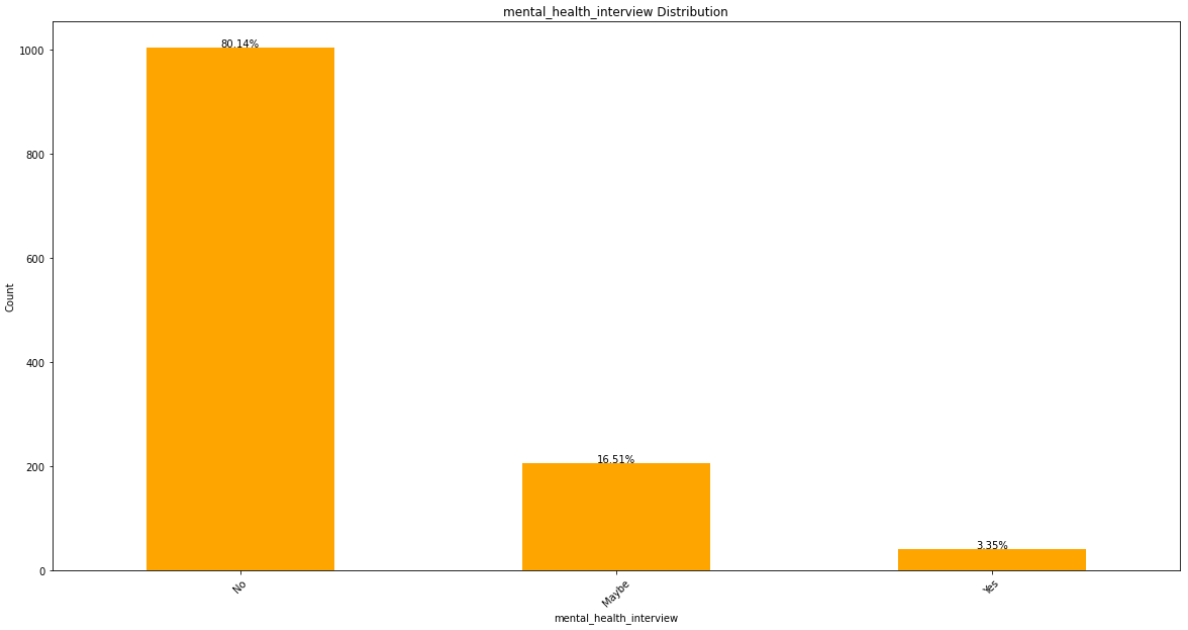

leave Distribution
Count
leave
Don't know
Somewhat easy
Very easy
Somewhat difficult
Very difficult
44.74%
21.21%
16.35%
9.97%
7.74%

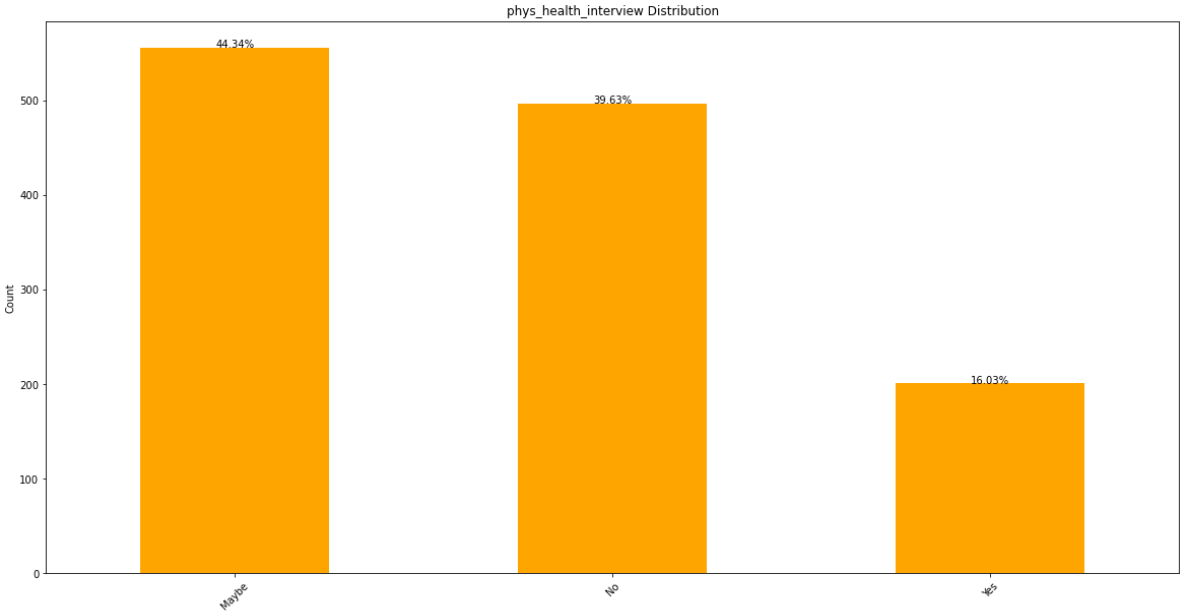

mental_health_consequence Distribution
Count
mental_health_consequence
No
Maybe
Yes
39.00%
37.96%
23.05%

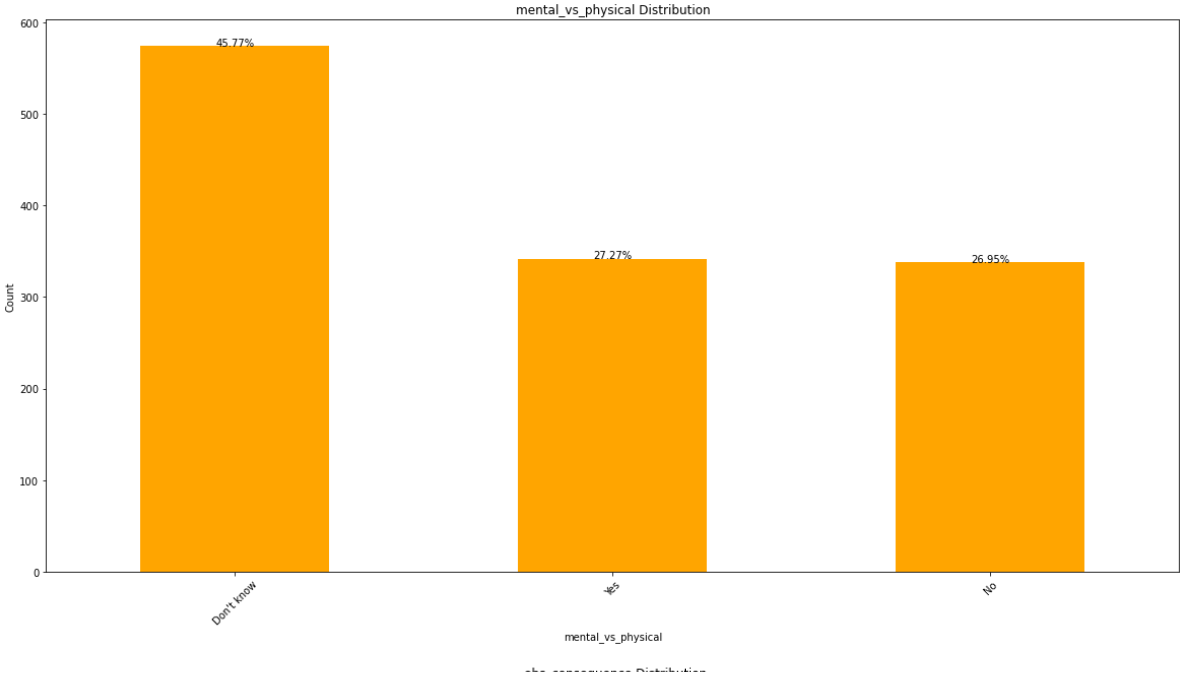

phys_health_consequence Distribution
Count
phys_health_consequence
No
Maybe
Yes
73.52%
21.77%
4.70%


coworkers Distribution
Count
61.56%
20.57%
17.86%
Some of them
No
Yes
coworkers


supervisor Distribution
Count
40.99%
31.10%
27.91%
Yes
No
Some of them
supervisor


mental_health_interview Distribution
Count
80.14%
16.51%
3.35%
No
Maybe
Yes
mental_health_interview

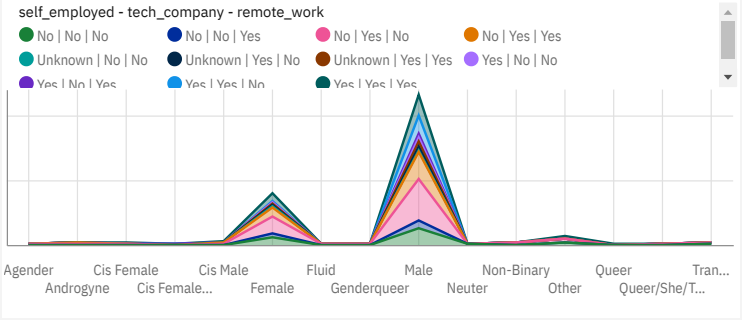

phys_health_interview Distribution
44.34%
39.63%
16.03%
Count
0
100
200
300
400
500
Maybe
No
Yes
phys_health_interview


mental_vs_physical Distribution
45.77%
27.27%
26.95%
Count
0
100
200
300
400
500
600
Don't know
Yes
No
mental_vs_physical

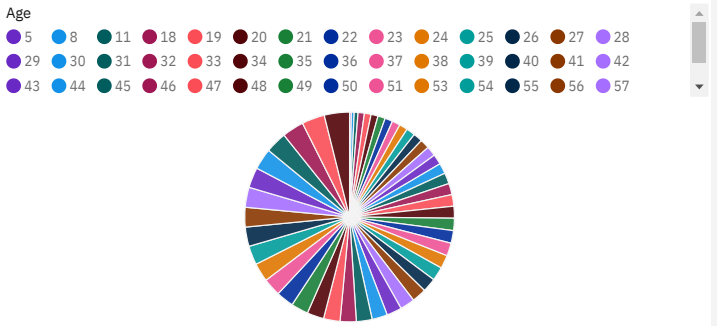
In [ ]:

Tab 1

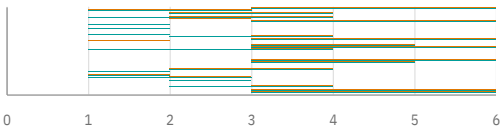
Country by Gender colored by self_employed, tech_company and remote_work



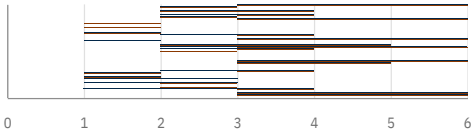
Age by Age



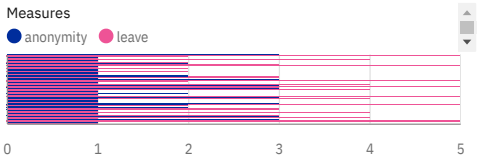
mental_health_consequence and phys_health_consequence by Country



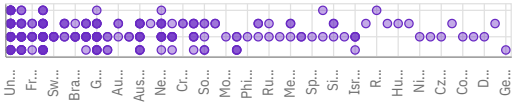
mental_health_interview and phys_health_interview by Country



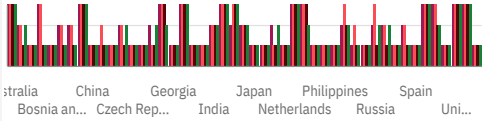
anonymity and leave by Country



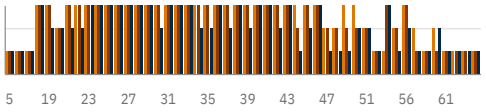
Country by tech_companyremote_work with points for no_employees



benefits, care_options, wellness_program and seek_help by Country

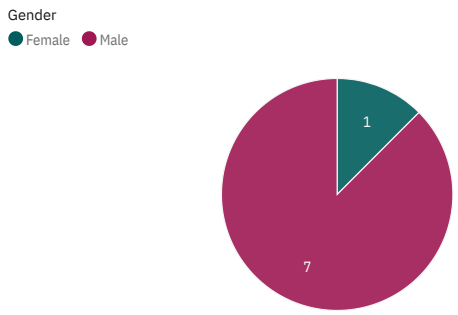


mental_health_consequence, phys_health_interview and mental_health_interview by Age

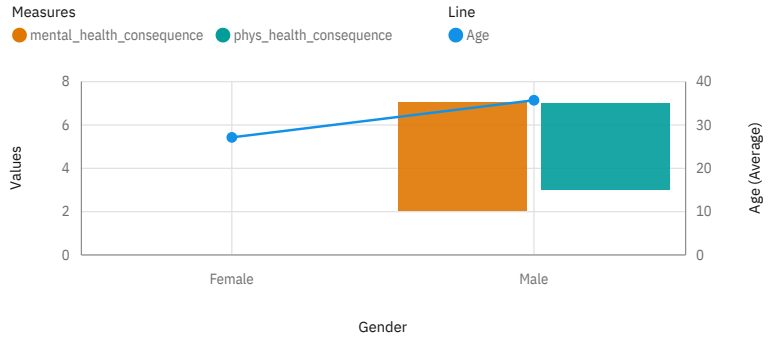


Tab 2

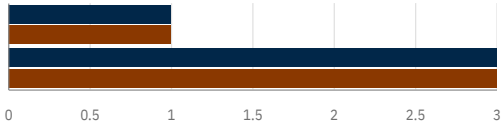
Gender by Gender



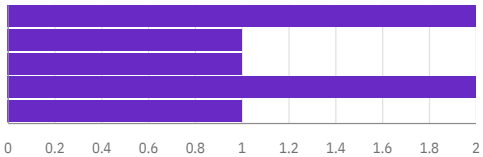
Age and mental_health_consequence and phys_health_consequence by Gender



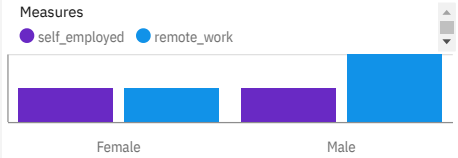
mental_health_interview and phys_health_interview by Gender



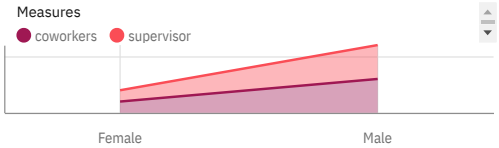
no_employees by Age



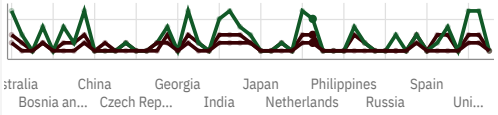
self_employed and remote_work by Gender



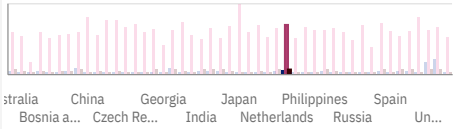
coworkers and supervisor by Gender



tech_company, no_employees and mental_vs_physical by Country



Gender, Age and mental_vs_physical by Country



Model Development and Evaluation

Table of Contents

1. Introduction
2. Model Development
3. Model Evaluation
4. Conclusion
5. Recommendations

1. Introduction

Phase 4 of our project focuses on model development and evaluation. In this phase, we implemented various classification algorithms to predict or classify specific outcomes related to our public health awareness campaign analysis. The algorithms we employed are as follows:

- Logistic Regression
- Decision Tree
- k-Nearest Neighbors (KNN)
- Random Forest
- AdaBoost
- Gradient Boosting

Our goal is to assess the performance of these models and determine which one is best suited for our campaign's objectives.

2. Model Development

In this section, we describe the development of each model:

Logistic Regression:

Logistic Regression is a simple yet effective model for binary classification. We implemented it to predict whether an individual is likely to be affected by mental health issues based on the features available in the dataset.

Decision Tree:

The Decision Tree model is a tree-like structure that recursively splits the data based on features. We employed it to identify key factors contributing to mental health issues in the tech industry.

k-Nearest Neighbors (KNN):

KNN is a distance-based classification method. We used it to classify individuals into groups based on similar attributes, which can be helpful in targeting specific demographics in the campaign.

Random Forest:

Random Forest is an ensemble learning technique that combines multiple Decision Trees. It was employed to improve the accuracy of our classification by reducing overfitting and enhancing generalization.

A**daBoost:**

AdaBoost is another ensemble learning method, which sequentially combines multiple weak learners to create a strong classifier. We utilized AdaBoost to enhance the performance of our classification model.

Gradient Boosting:

Gradient Boosting is yet another ensemble technique that builds models in a sequential manner. It was used to improve the predictive accuracy and precision of our campaign's outcome.

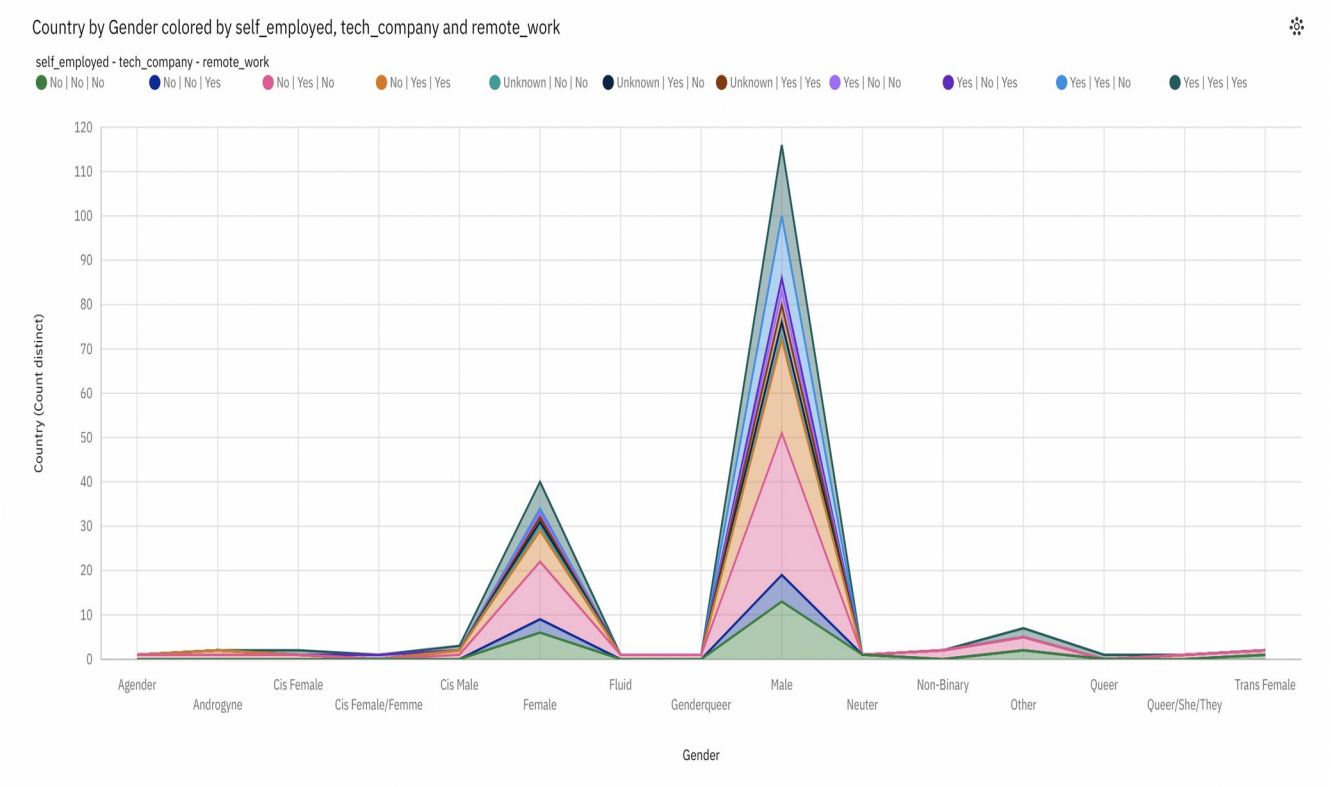
3. Model Evaluation

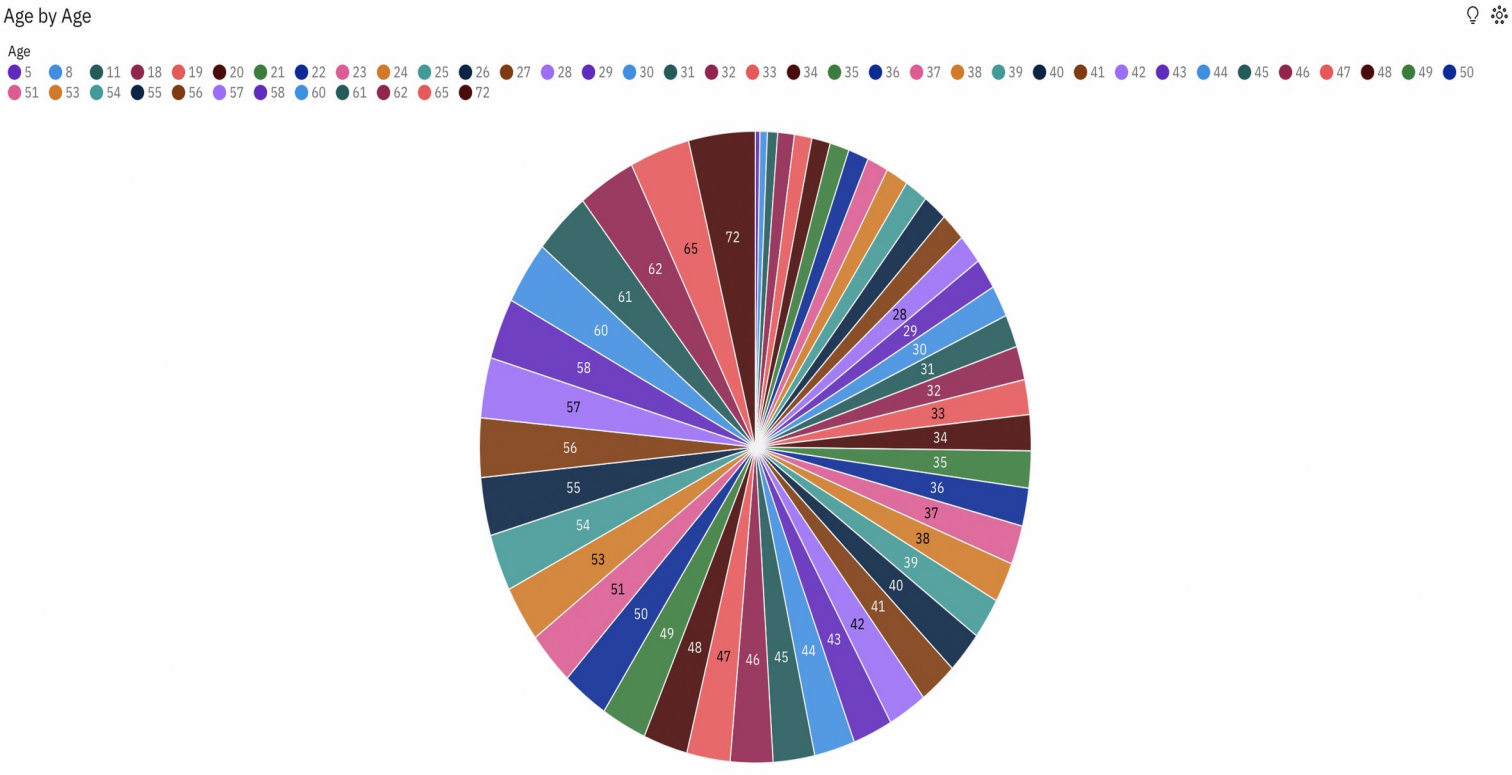
To evaluate the performance of these models, we employed several evaluation metrics, including but not limited to:

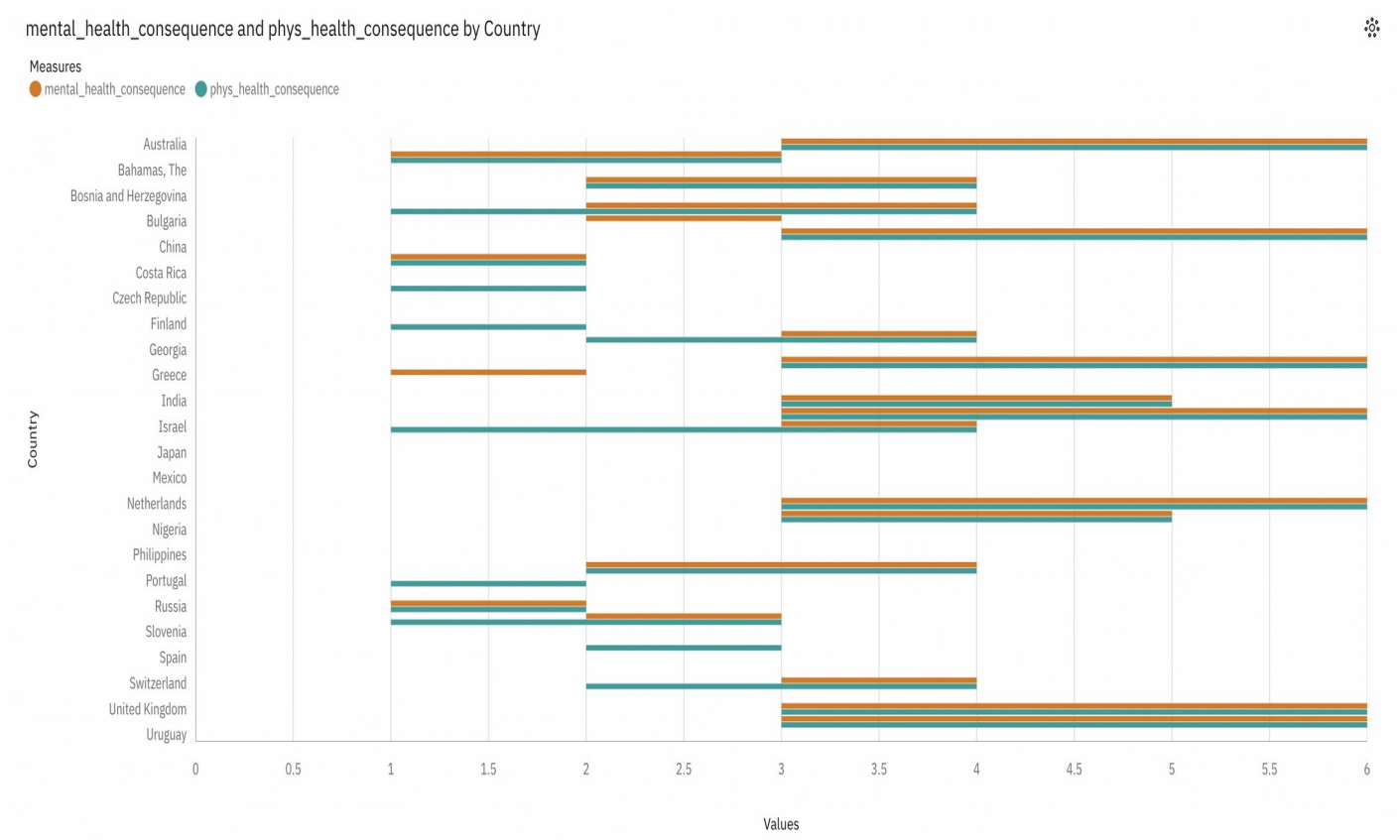
- Accuracy
- Precision
- Recall
- F1 Score
- ROC-AUC
- Confusion Matrix

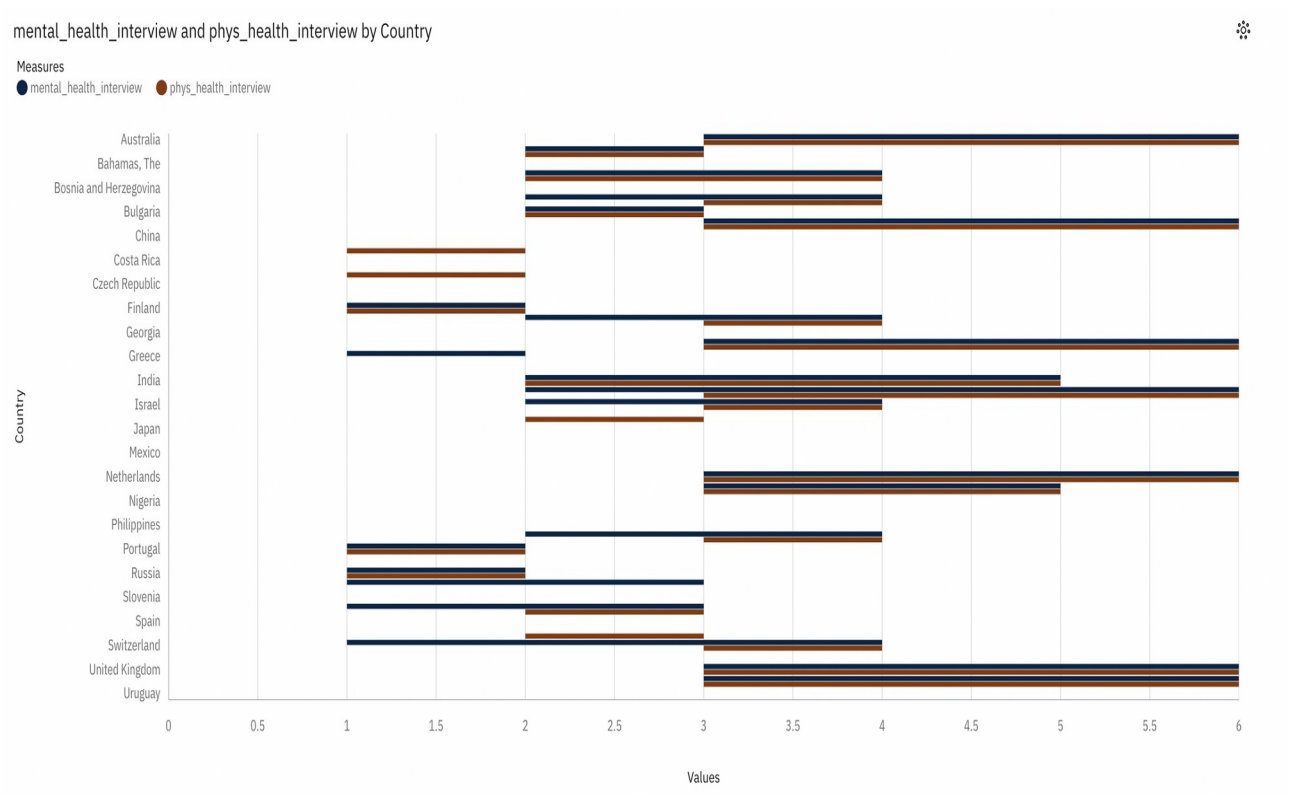
The results of the model evaluation are summarized in the following table:

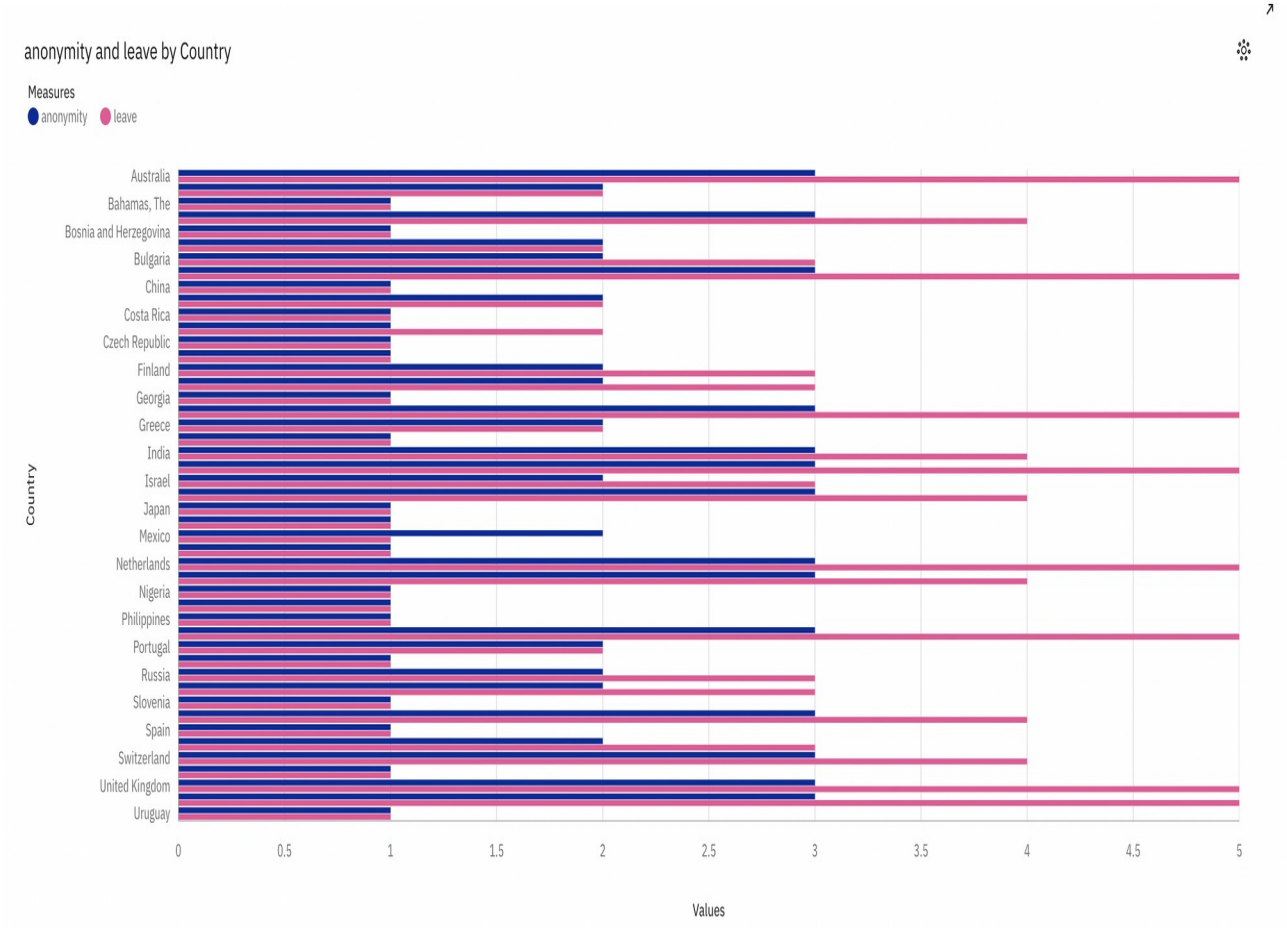
	method	cv score	mean score	std score	recall score
0	Logistic Regression	[0.80899, 0.88764, 0.88764, 0.88636, 0.875]	0.869127	0.030442	0.863158
1	Decision Tree Classifier	[0.76404, 0.68539, 0.83146, 0.84091, 0.79545]	0.783453	0.056111	0.747368
2	KNN Classifier	[0.57303, 0.67416, 0.70787, 0.65909, 0.65909]	0.654648	0.044527	0.605263
3	Random Forest Classifier	[0.77528, 0.83146, 0.88764, 0.90909, 0.86364]	0.853422	0.046824	0.810526
4	Ada Boost Classifier	[0.80899, 0.85393, 0.92135, 0.84091, 0.85227]	0.855490	0.036674	0.857895
5	Gradient Boosting Classifier	[0.78652, 0.80899, 0.89888, 0.89773, 0.86364]	0.851149	0.045953	0.800000

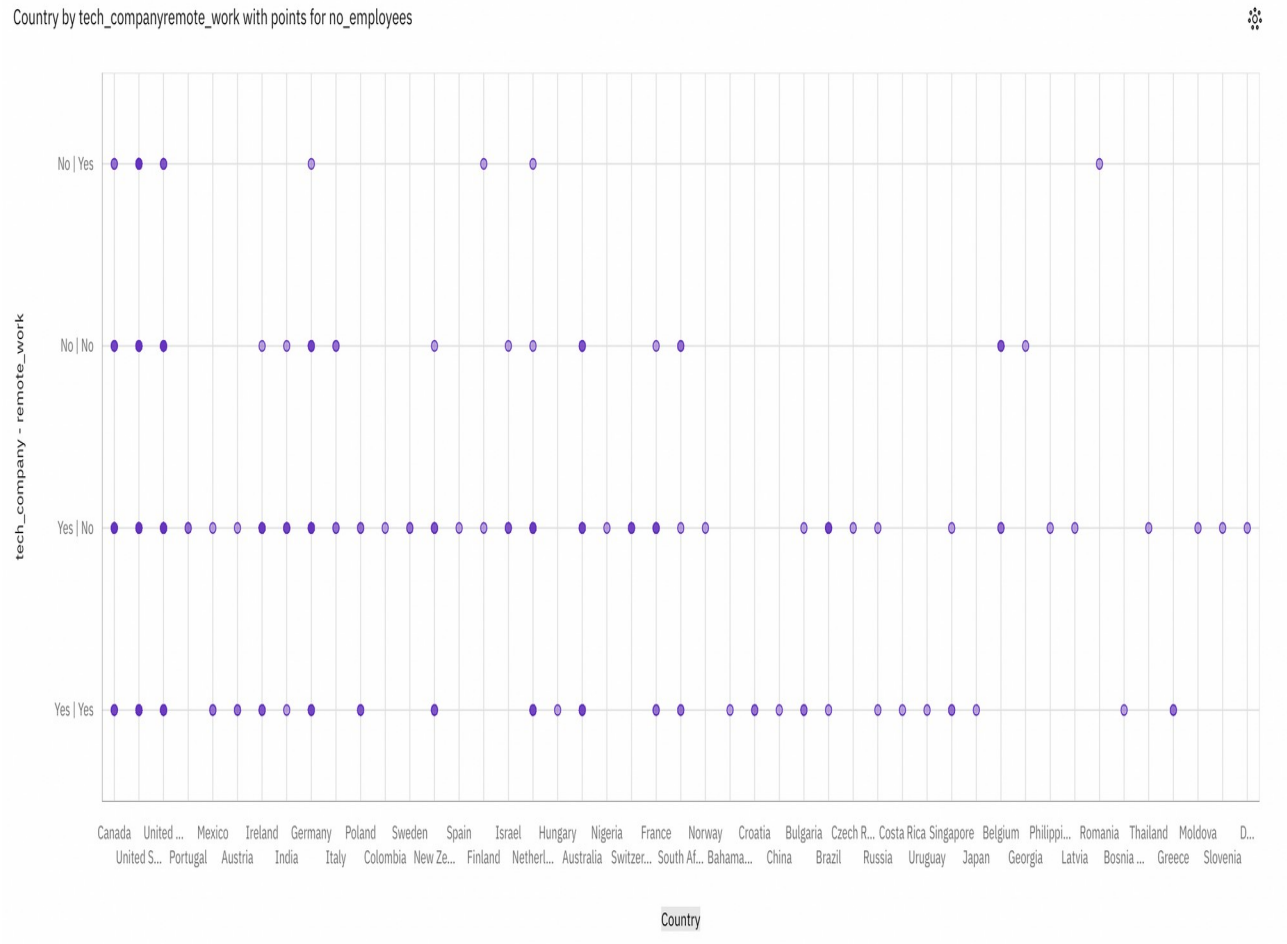

Country by Gender colored by self_employed, tech_company and remote_work
self_employed - tech_company - remote_work
No | No | No
No | No | Yes
No | Yes | No
No | Yes | Yes
Unknown | No | No
Unknown | Yes | No
Unknown | Yes | Yes
Yes | No | No
Yes | No | Yes
Yes | Yes | No
Yes | Yes | Yes
Country (Count distinct)
120
110
100
90
80
70
60
50
40
30
20
10
0
Agender
Androgyne
Cis Female
Cis Female/Femme
Cis Male
Female
Fluid
Genderqueer
Male
Neuter
Non-Binary
Other
Queer
Queer/She/They
Trans Female
Gender

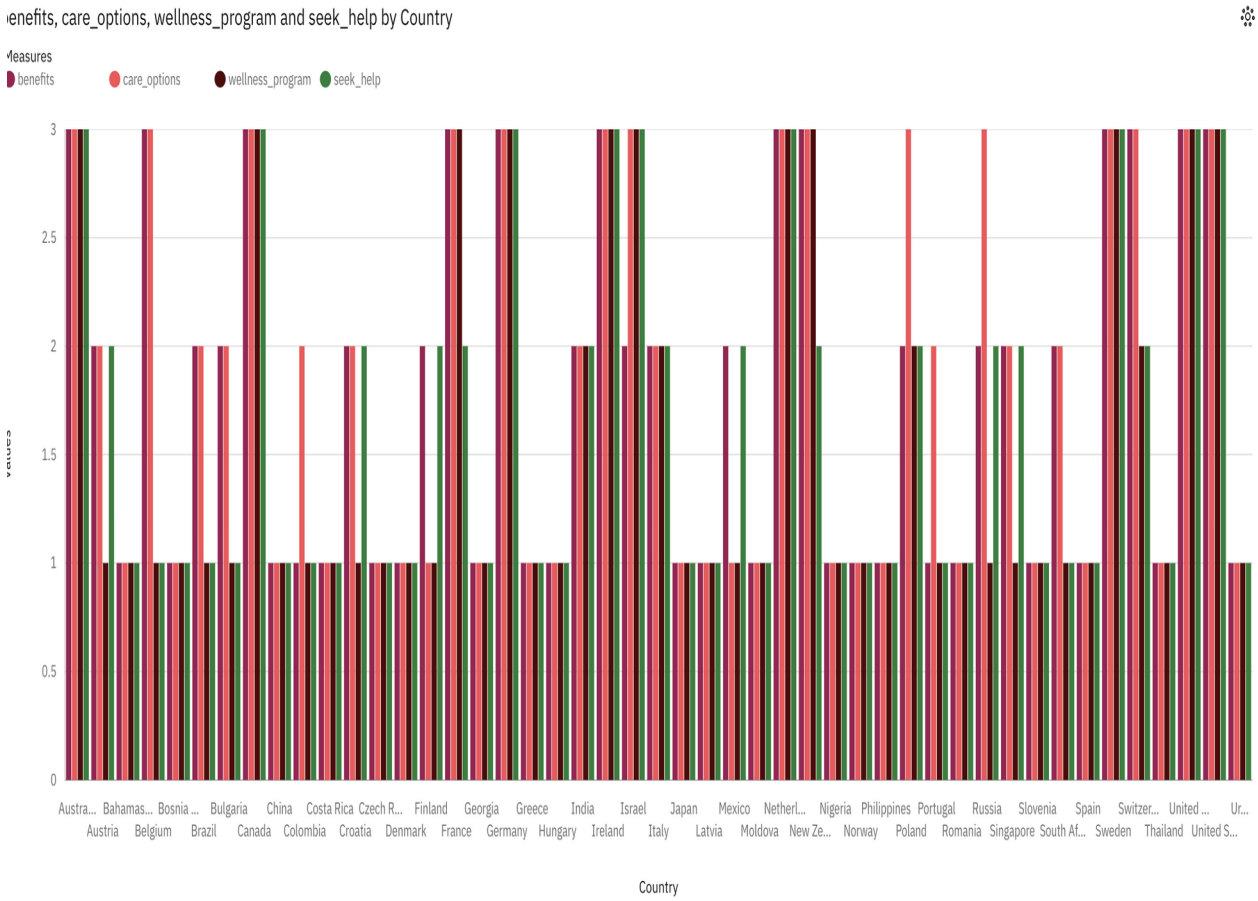

Age by Age
Age
5 8 11 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
51 53 54 55 56 57 58 60 61 62 65 72
72
65
62
61
60
58
57
56
55
54
53
51
50
49
48
47
46
45
44
43
42
41
40
39
38
37
36
35
34
33
32
31
30
29
28

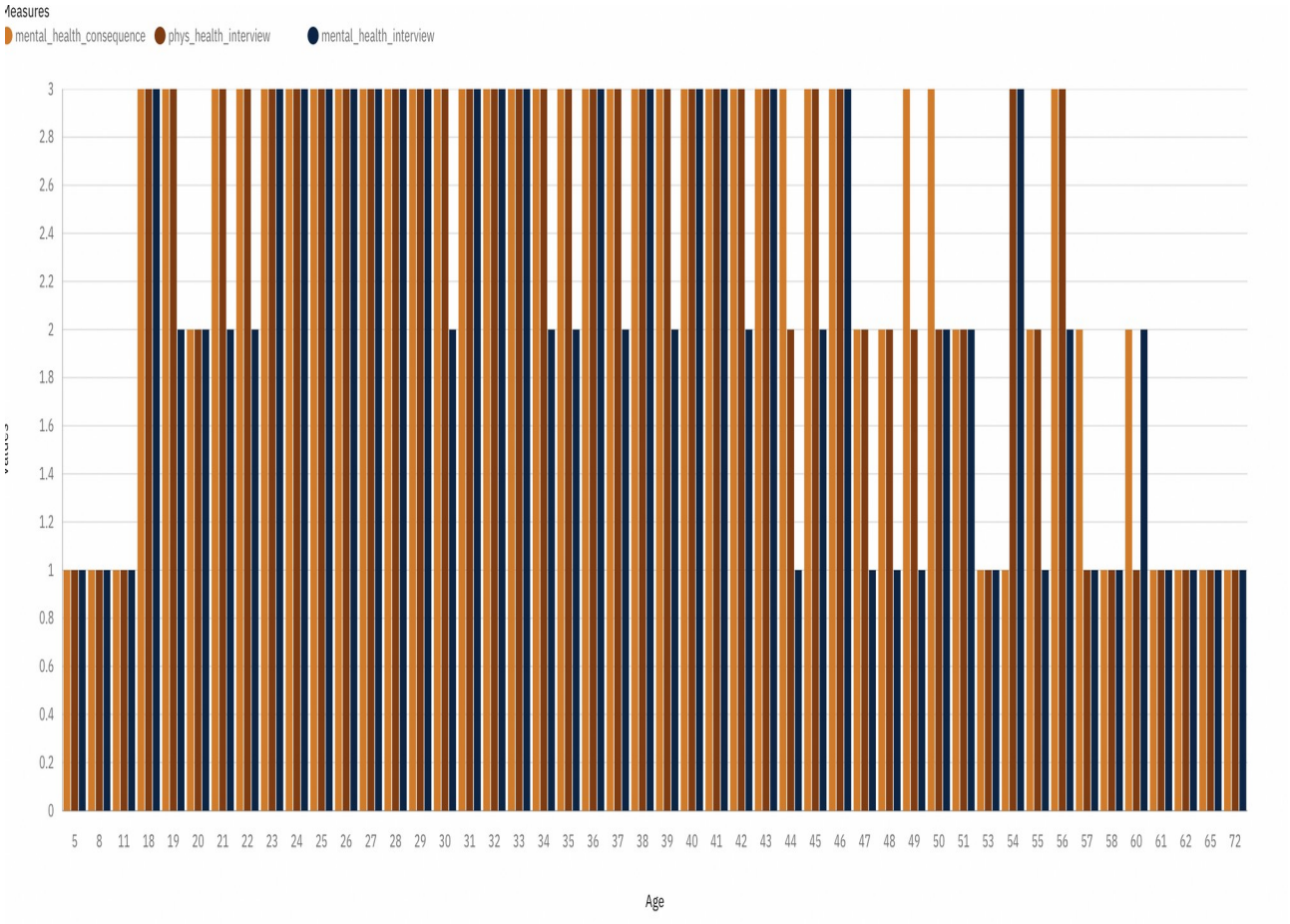

mental_health_consequence and phys_health_consequence by Country
Measures
mental_health_consequence
phys_health_consequence
Australia
Bahamas, The
Bosnia and Herzegovina
Bulgaria
China
Costa Rica
Czech Republic
Finland
Georgia
Greece
India
Israel
Japan
Mexico
Netherlands
Nigeria
Philippines
Portugal
Russia
Slovenia
Spain
Switzerland
United Kingdom
Uruguay
Country
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
6
Values


mental_health_interview and phys_health_interview by Country
Measures
mental_health_interview
phys_health_interview
Australia
Bahamas, The
Bosnia and Herzegovina
Bulgaria
China
Costa Rica
Czech Republic
Finland
Georgia
Greece
India
Israel
Japan
Mexico
Netherlands
Nigeria
Philippines
Portugal
Russia
Slovenia
Spain
Switzerland
United Kingdom
Uruguay
Country
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
6
Values

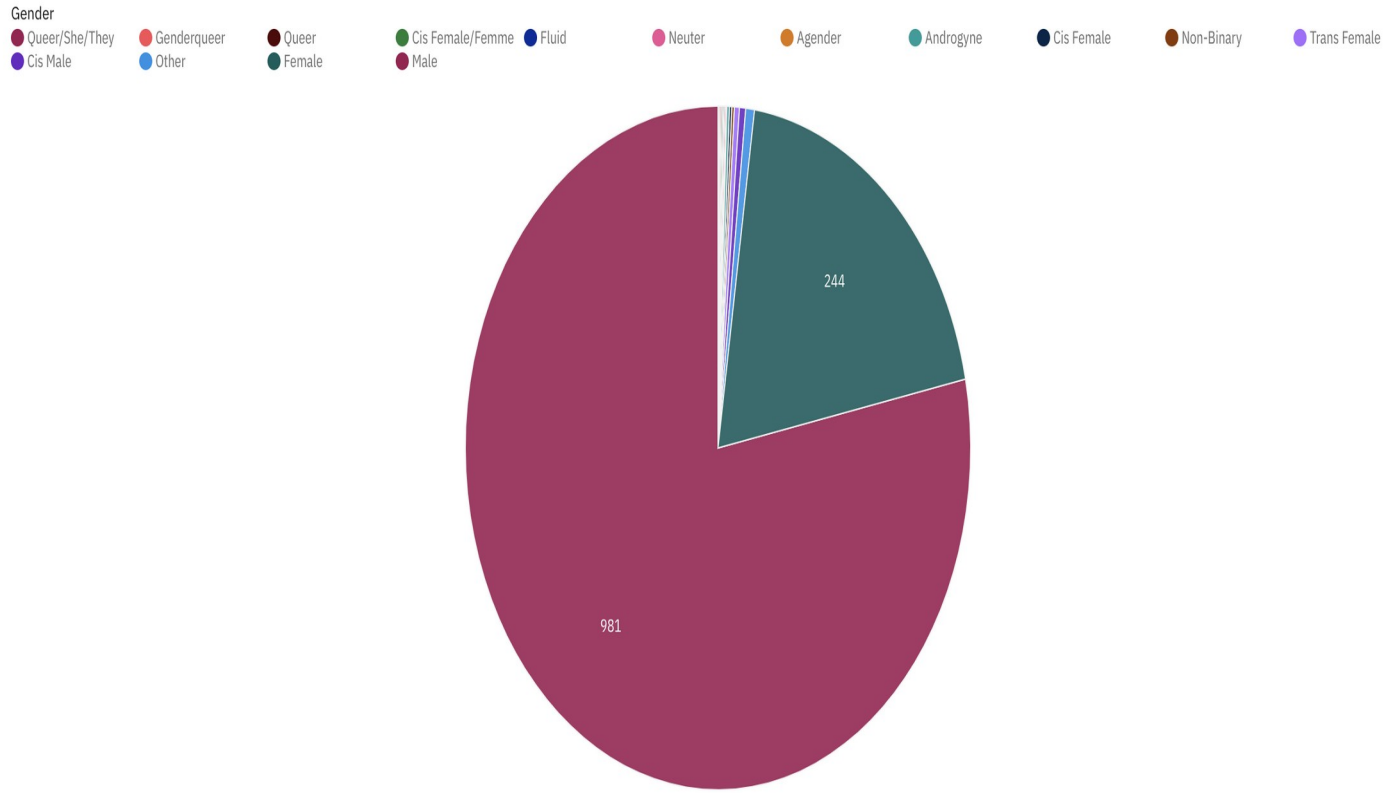

anonymity and leave by Country
Measures
anonymity
leave
Australia
Bahamas, The
Bosnia and Herzegovina
Bulgaria
China
Costa Rica
Czech Republic
Finland
Georgia
Greece
India
Israel
Japan
Mexico
Netherlands
Nigeria
Philippines
Portugal
Russia
Slovenia
Spain
Switzerland
United Kingdom
Uruguay
Country
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
Values


Country by tech_companyremote_work with points for no_employees
No | Yes
No | No
Yes | No
Yes | Yes
tech_company - remote_work
Canada
United S...
United ...
Portugal
Mexico
Austria
Ireland
India
Germany
Italy
Poland
Colombia
Sweden
New Ze...
Spain
Finland
Israel
Netherl...
Hungary
Australia
Nigeria
Switzer...
France
South Af...
Norway
Bahama...
Croatia
China
Bulgaria
Brazil
Czech R...
Russia
Costa Rica
Uruguay
Singapore
Japan
Belgium
Georgia
Philippi...
Latvia
Romania
Bosnia ...
Thailand
Greece
Moldova
Slovenia
D...
Country

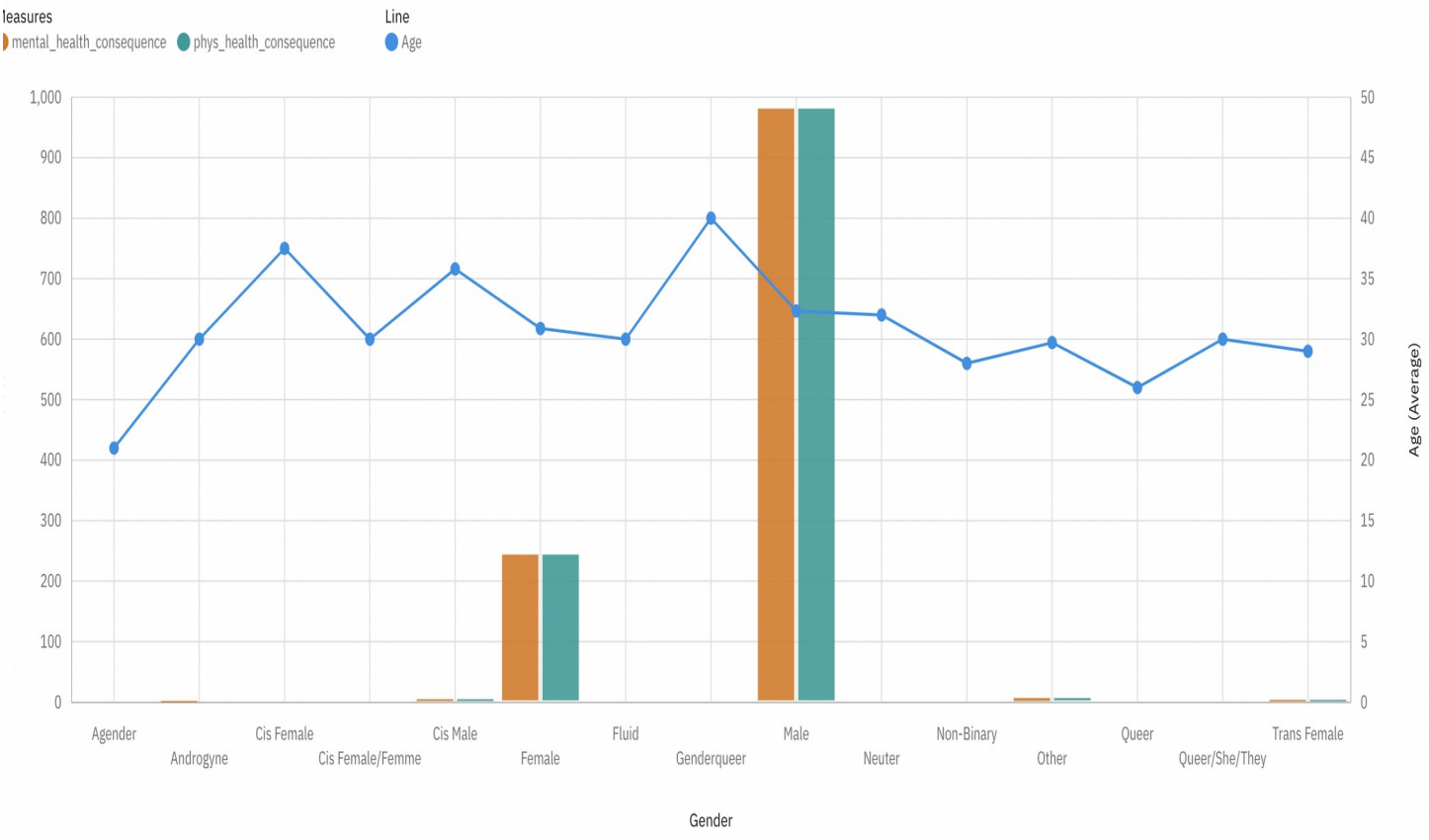

enefits, care_options, wellness_program and seek_help by Country
Measures
benefits
care_options
wellness_program
seek_help
3
2.5
2
1.5
1
0.5
0
Austra...
Bahamas...
Bosnia ...
Bulgaria
China
Costa Rica
Czech R...
Finland
Georgia
Greece
India
Israel
Japan
Mexico
Netherl...
Nigeria
Philippines
Portugal
Russia
Slovenia
Spain
Switzer...
United ...
Ur...
Austria
Belgium
Brazil
Canada
Colombia
Croatia
Denmark
France
Germany
Hungary
Ireland
Italy
Latvia
Moldova
New Ze...
Norway
Poland
Romania
Singapore
South Af...
Sweden
Thailand
United S...
Country

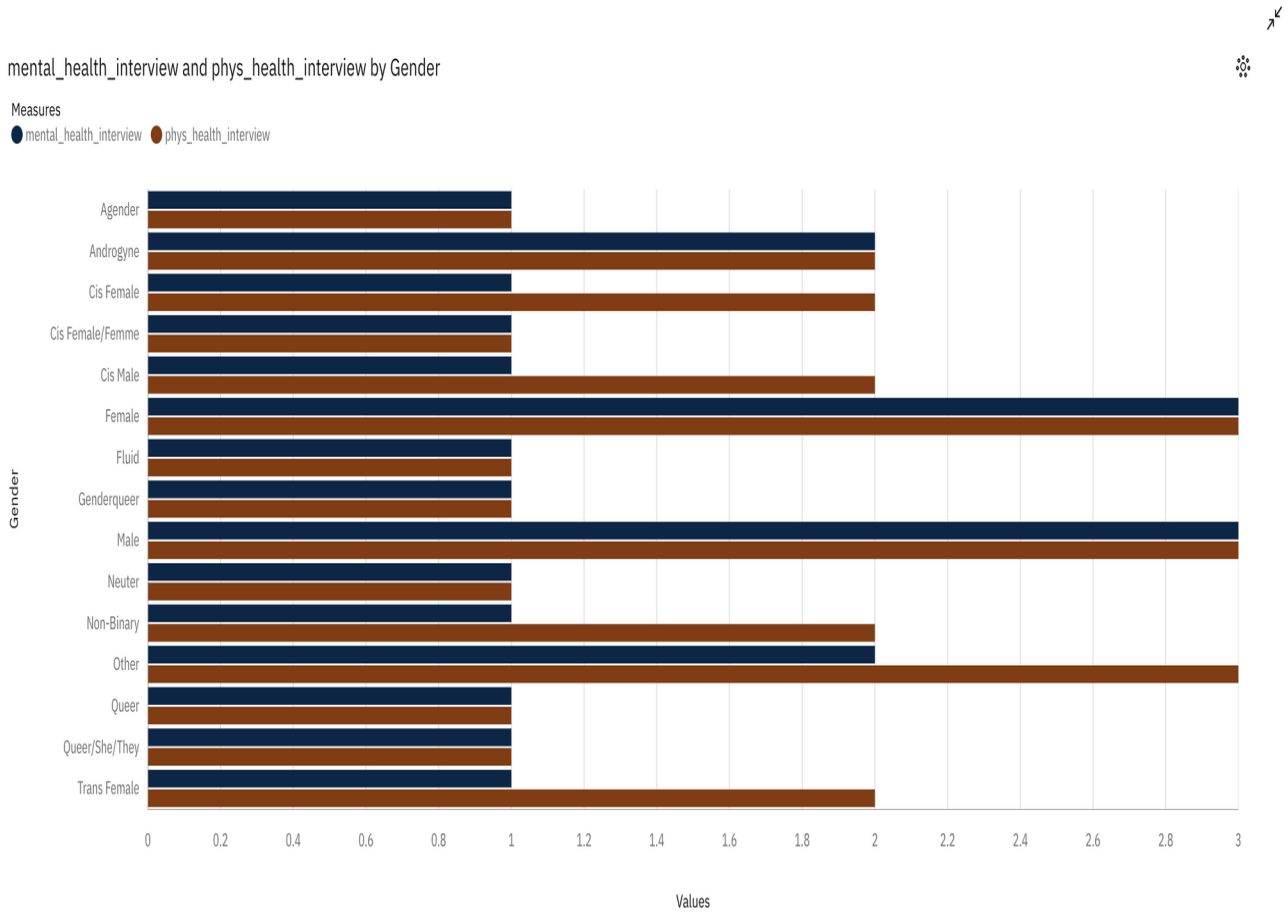

Measures
mental_health_consequence
phys_health_interview
mental_health_interview
3
2.8
2.6
2.4
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
5
8
11
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
53
54
55
56
57
58
60
61
62
65
72
Age

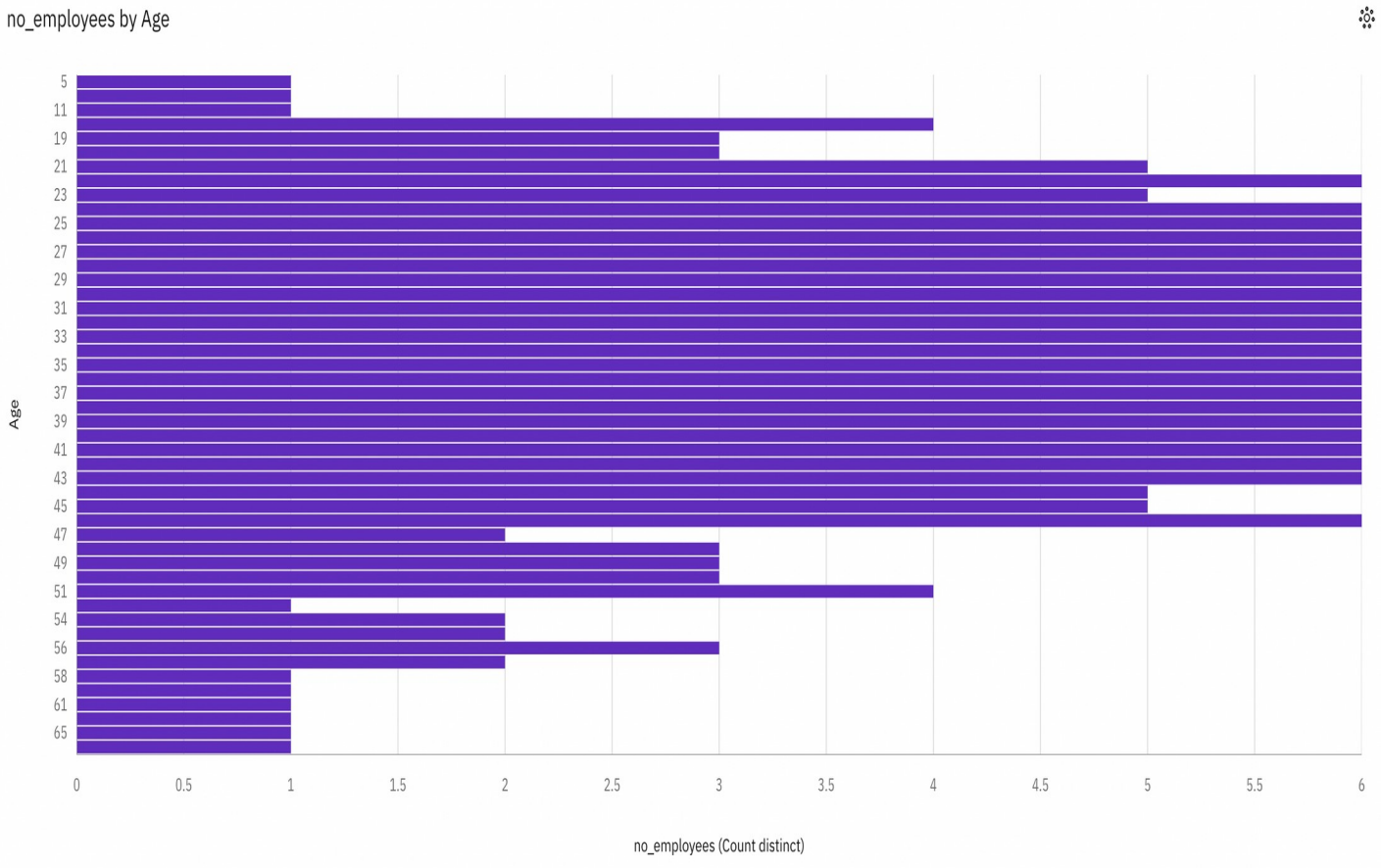
Gender by Gender

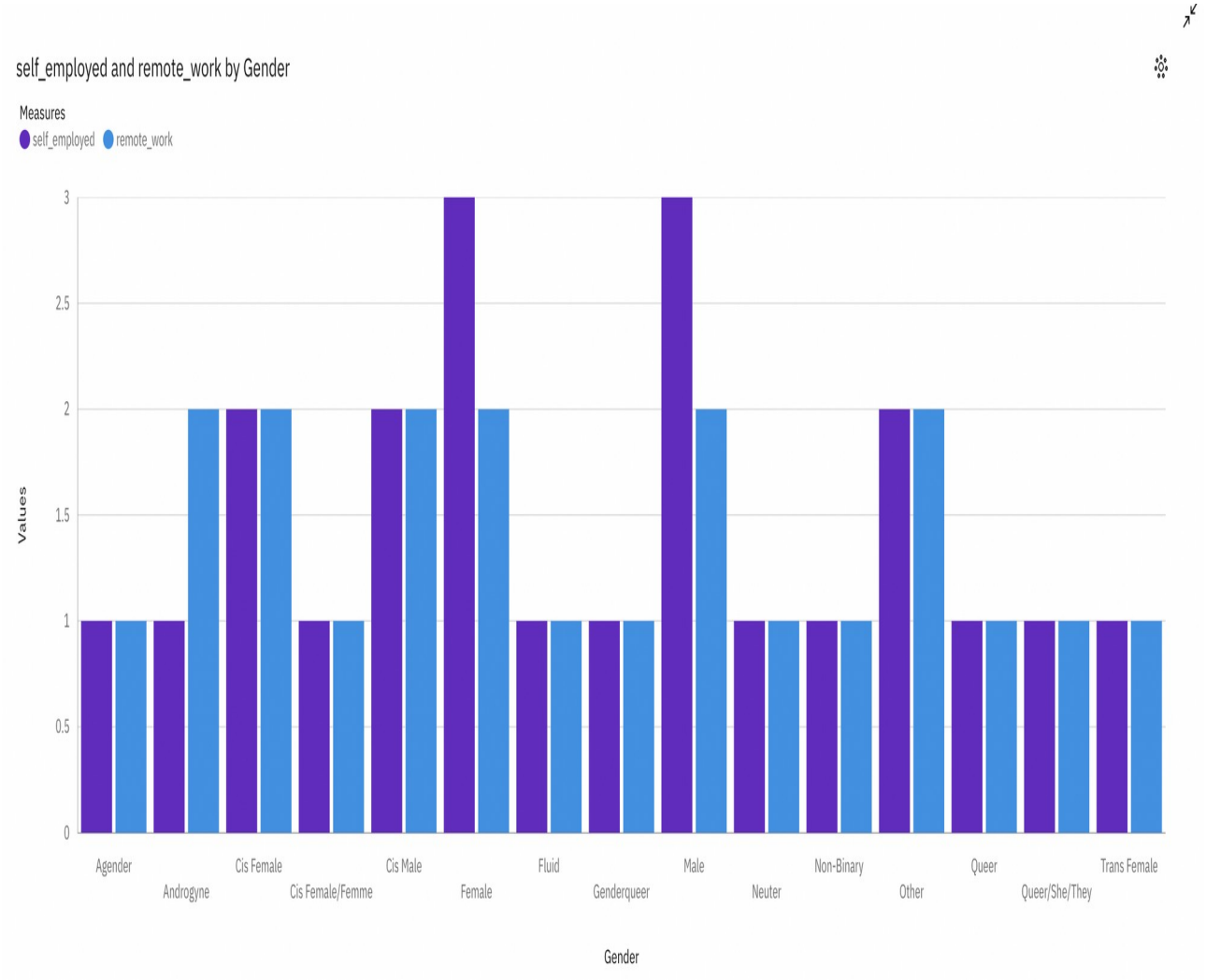


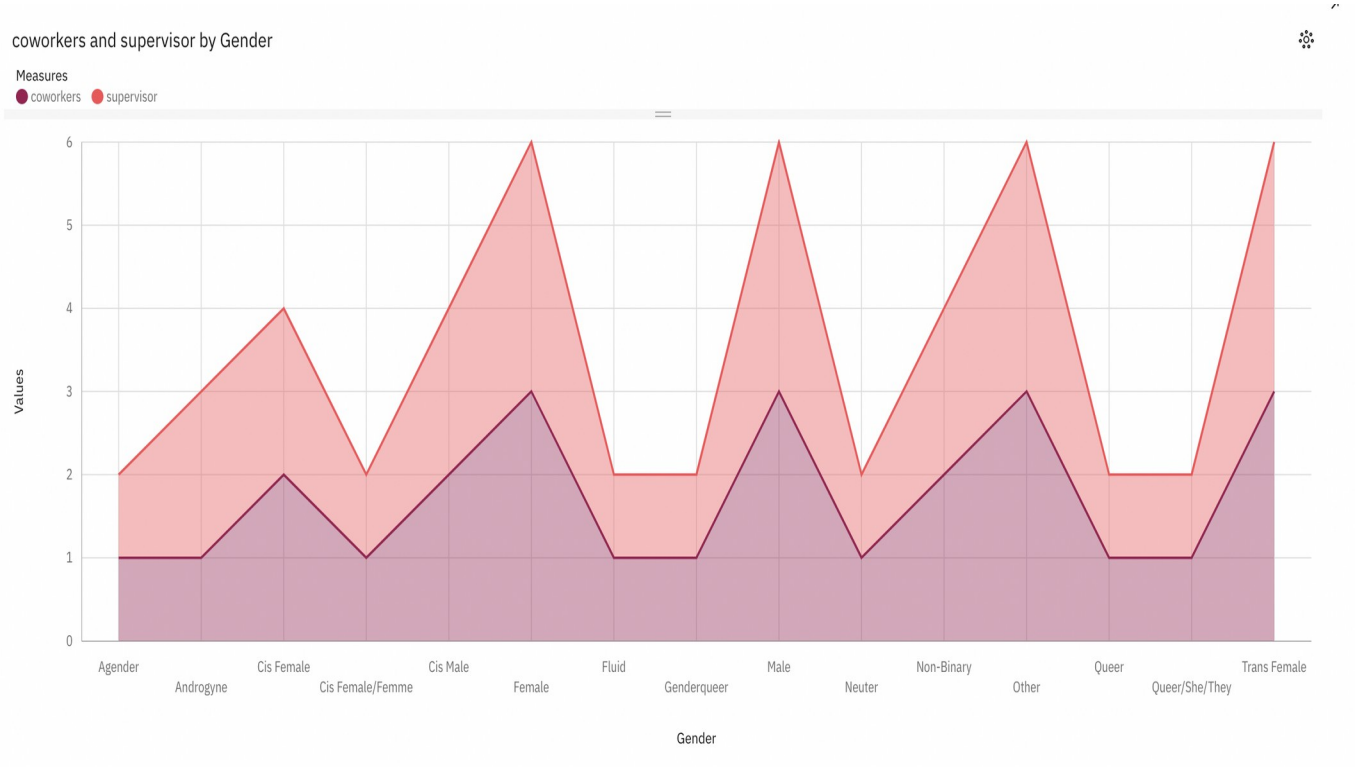
ge and mental_health_consequence and phys_health_consequence by Gender

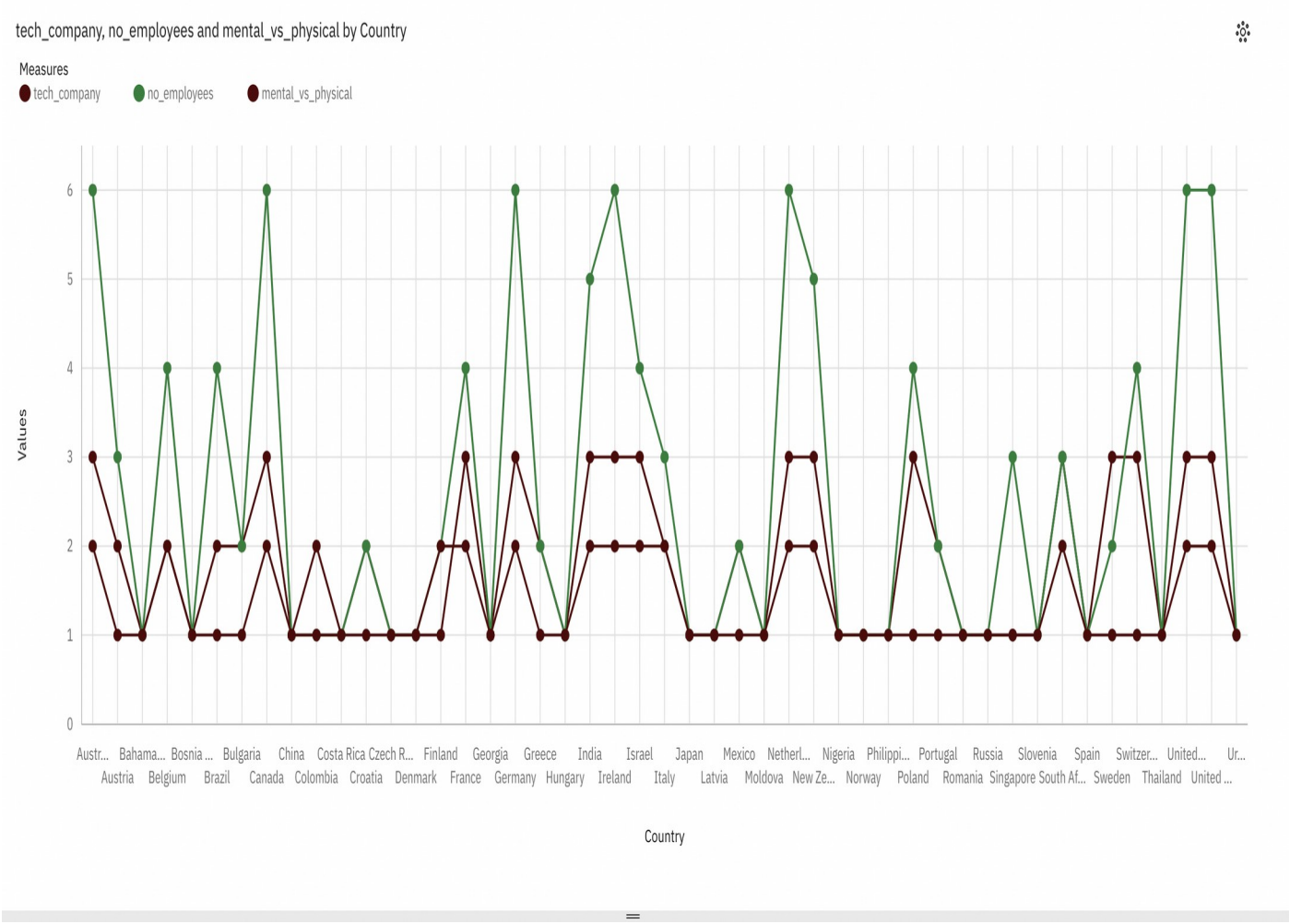


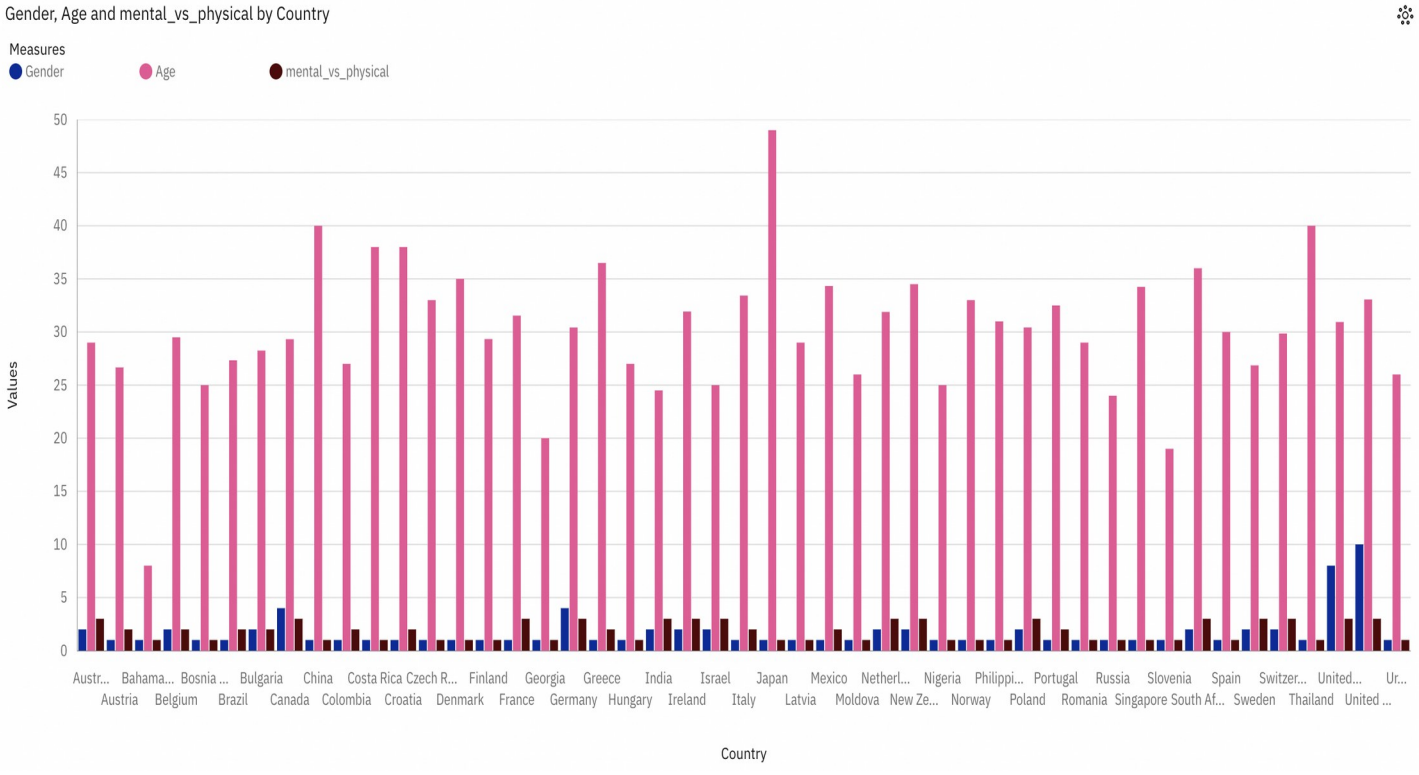

mental_health_interview and phys_health_interview by Gender
Measures
mental_health_interview
phys_health_interview
Agender
Androgyne
Cis Female
Cis Female/Femme
Cis Male
Female
Fluid
Genderqueer
Male
Neuter
Non-Binary
Other
Queer
Queer/She/They
Trans Female
Gender
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
2.2
2.4
2.6
2.8
3
Values


no_employees by Age
5
11
19
21
23
25
27
29
31
33
35
37
39
41
43
45
47
49
51
54
56
58
61
65
Age
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
6
no_employees (Count distinct)


self_employed and remote_work by Gender
Measures
self_employed
remote_work
Values
3
2.5
2
1.5
1
0.5
0
Agender
Androgyne
Cis Female
Cis Female/Femme
Cis Male
Female
Fluid
Genderqueer
Male
Neuter
Non-Binary
Other
Queer
Queer/She/They
Trans Female
Gender


coworkers and supervisor by Gender
Measures
coworkers
supervisor
Values
6
5
4
3
2
1
0
Agender
Androgyne
Cis Female
Cis Female/Femme
Cis Male
Female
Fluid
Genderqueer
Male
Neuter
Non-Binary
Other
Queer
Queer/She/They
Trans Female
Gender


tech_company, no_employees and mental_vs_physical by Country
Measures
tech_company
no_employees
mental_vs_physical
Values
0
1
2
3
4
5
6
Austr... Bahama... Bosnia ... Bulgaria China Costa Rica Czech R... Finland Georgia Greece India Israel Japan Mexico Netherl... Nigeria Philippi... Portugal Russia Slovenia Spain Switzer... United... Ur...
Austria Belgium Brazil Canada Colombia Croatia Denmark France Germany Hungary Ireland Italy Latvia Moldova New Ze... Norway Poland Romania Singapore South Af... Sweden Thailand United ...
Country


Gender, Age and mental_vs_physical by Country
Measures
Gender
Age
mental_vs_physical
Values
0
5
10
15
20
25
30
35
40
45
50
Austr... Bahama... Bosnia ... Bulgaria China Costa Rica Czech R... Finland Georgia Greece India Israel Japan Mexico Netherl... Nigeria Philippi... Portugal Russia Slovenia Spain Switzer... United... Ur...
Austria Belgium Brazil Canada Colombia Croatia Denmark France Germany Hungary Ireland Italy Latvia Moldova New Ze... Norway Poland Romania Singapore South Af... Sweden Thailand United ...
Country

In [ ]:

In [ ]:
```
PHASE 4
```

In [ ]:

In [ ]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")

# PreProcessing
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.pipeline import Pipeline
from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import SimpleImputer, IterativeImputer
from sklearn.preprocessing import MinMaxScaler


# Splitting Data
from sklearn.model_selection import train_test_split, StratifiedKFold, cross_val_score

# Modeling
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, recall_score, plot_roc_curve
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier, GradientBoostingClassifier


# Tuning
from sklearn.model_selection import GridSearchCV
```

In [ ]:

In [ ]:
```
Data Cleaning
```

In [ ]:

In [67]:
```
mh = pd.read_csv('cleaned_survey.csv')
mh
```

Out[67]:

	Timestamp	Age	Gender	Country	state	self_employed	family_history	treatment	work_interfere	no_employees	...	leave	ment
0	2014-08-27 11:29:31	37	Female	United States	IL	Unknown	No	Yes	Often	6-25	...	Somewhat easy	
1	2014-08-27 11:29:37	44	Male	United States	IN	Unknown	No	No	Rarely	More than 1000	...	Don't know	
2	2014-08-27 11:29:44	32	Male	Canada	Unknown	Unknown	No	No	Rarely	6-25	...	Somewhat difficult	
3	2014-08-27 11:29:46	31	Male	United Kingdom	Unknown	Unknown	Yes	Yes	Often	26-100	...	Somewhat difficult	
4	2014-08-27 11:30:22	31	Male	United States	TX	Unknown	No	No	Never	100-500	...	Don't know	
...	...	...	...	...	...	...	...	...	...	...	...	...	
1249	2015-09-12 11:17:21	26	Male	United Kingdom	Unknown	No	No	Yes	Unknown	26-100	...	Somewhat easy	
1250	2015-09-26 01:07:35	32	Male	United States	IL	No	Yes	Yes	Often	26-100	...	Somewhat difficult	
1251	2015-11-07 12:36:58	34	Male	United States	CA	No	Yes	Yes	Sometimes	More than 1000	...	Somewhat difficult	
1252	2015-11-30 21:25:06	46	Female	United States	NC	No	No	No	Unknown	100-500	...	Don't know	
1253	2016-02-01 23:04:31	25	Male	United States	IL	No	Yes	Yes	Sometimes	26-100	...	Don't know	

1254 rows × 27 columns

In [68]:
```
mh.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1254 entries, 0 to 1253
```

```
Data columns (total 27 columns):
 #   Column                     Non-Null Count  Dtype 
---  ------                     --------------  ----- 
 0   Timestamp                  1254 non-null   object
 1   Age                        1254 non-null   int64 
 2   Gender                     1254 non-null   object
 3   Country                    1254 non-null   object
 4   state                      1254 non-null   object
 5   self_employed              1254 non-null   object
 6   family_history             1254 non-null   object
 7   treatment                  1254 non-null   object
 8   work_interfere             1254 non-null   object
 9   no_employees               1254 non-null   object
 10  remote_work                1254 non-null   object
 11  tech_company               1254 non-null   object
 12  benefits                   1254 non-null   object
 13  care_options               1254 non-null   object
 14  wellness_program           1254 non-null   object
 15  seek_help                  1254 non-null   object
 16  anonymity                  1254 non-null   object
 17  leave                      1254 non-null   object
 18  mental_health_consequence  1254 non-null   object
 19  phys_health_consequence    1254 non-null   object
 20  coworkers                  1254 non-null   object
 21  supervisor                 1254 non-null   object
 22  mental_health_interview    1254 non-null   object
 23  phys_health_interview      1254 non-null   object
 24  mental_vs_physical         1254 non-null   object
 25  obs_consequence            1254 non-null   object
 26  comments                   1254 non-null   object
dtypes: int64(1), object(26)
memory usage: 264.6+ KB
```

In [69]:
```
mh.isna().sum()/len(mh.index)*100
```

Out[69]:
```
Timestamp                  0.0
Age                        0.0
Gender                     0.0
Country                    0.0
state                      0.0
self_employed              0.0
family_history             0.0
treatment                  0.0
work_interfere             0.0
no_employees               0.0
remote_work                0.0
tech_company               0.0
```

```
benefits                    0.0
care_options                0.0
wellness_program            0.0
seek_help                   0.0
anonymity                   0.0
leave                       0.0
mental_health_consequence   0.0
phys_health_consequence     0.0
coworkers                   0.0
supervisor                  0.0
mental_health_interview     0.0
phys_health_interview       0.0
mental_vs_physical          0.0
obs_consequence             0.0
comments                    0.0
dtype: float64
```

In [70]:
```
mh.drop(columns=['Timestamp', 'Country', 'state', 'comments'], inplace = True)
```

In [71]:
```
mh.rename({'self_employed' : 'Self_Employed', 'family_history' : 'Family_History',
           'treatment' : 'Treatment', 'work_interfere' : 'Work_Interfere',
           'no_employees': 'Employee_Numbers', 'remote_work': 'Remote_Work', 'tech_company': 'Tech_Company',
           'benefits': 'Benefits', 'care_options': 'Care_Options', 'wellness_program': 'Wellness_Program',
           'seek_help': 'Seek_Help', 'anonymity': 'Anonymity', 'leave': 'Medical_Leave',
           'mental_health_consequence': 'Mental_Health_Consequence',
           'phys_health_consequence': 'Physical_Health_Consequence', 'coworkers': 'Coworkers',
           'supervisor': 'Supervisor', 'mental_health_interview': 'Mental_Health_Interview',
           'phys_health_interview': 'Physical_Health_Interview', 'mental_vs_physical': 'Mental_VS_Physical',
           'obs_consequence': 'Observed_Consequence'} , inplace = True , axis = 1)
```

In [72]:
```
mh['Age'].unique()
```

Out[72]:
```
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, 43, 56, 60, 54, 55, 48,
       20, 57, 58, 47, 62, 51, 65, 49,  5, 53, 61,  8, 11, 72])
```

In [73]:
```
mh['Age'].replace([mh['Age'][mh['Age'] < 15]], np.nan, inplace = True)
mh['Age'].replace([mh['Age'][mh['Age'] > 100]], np.nan, inplace = True)

mh['Age'].unique()
```

Out[73]:
```
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., 43., 56., 60., 54., 55., 48., 20., 57., 58., 47., 62.,
       51., 65., 49., nan, 53., 61., 72.])
```

In [74]:
```
mh['Gender'].unique()
```

Out[74]:
```
array(['Female', 'Male', 'Trans Female', 'Cis Female', 'Cis Male',
       'Queer/She/They', 'Non-Binary', 'Other', 'Fluid', 'Genderqueer',
       'Androgyne', 'Agender', 'Cis Female/Femme', 'Neuter', 'Queer'],
      dtype=object)
```

In [75]:
```
mh['Gender'].replace(['Male ', 'male', 'M', 'm', 'Male', 'Cis Male',
                      'Man', 'cis male', 'Mail', 'Male-ish', 'Male (CIS)',
                       'Cis Man', 'msle', 'Malr', 'Mal', 'maile', 'Make',], 'Male', inplace = True)

mh['Gender'].replace(['Female ', 'female', 'F', 'f', 'Woman', 'Female',
                      'femail', 'Cis Female', 'cis-female/femme', 'Femake', 'Female (cis)',
                      'woman',], 'Female', inplace = True)

mh["Gender"].replace(['Female (trans)', 'queer/she/they', 'non-binary',
                      'fluid', 'queer', 'Androgyne', 'Trans-female', 'male leaning androgynous',
                       'Agender', 'A little about you', 'Nah', 'All',
                       'ostensibly male, unsure what that really means',
                       'Genderqueer', 'Enby', 'p', 'Neuter', 'something kinda male?',
                       'Guy (-ish) ^_^', 'Trans woman',], 'Queer', inplace = True)
```

In [76]:
```
mh['Gender'].value_counts()
```

Out[76]:
```
Male                986
Female              246
Other                 7
Queer                 6
Trans Female          4
Non-Binary            2
Cis Female/Femme      1
Fluid                 1
Queer/She/They        1
Name: Gender, dtype: int64
```

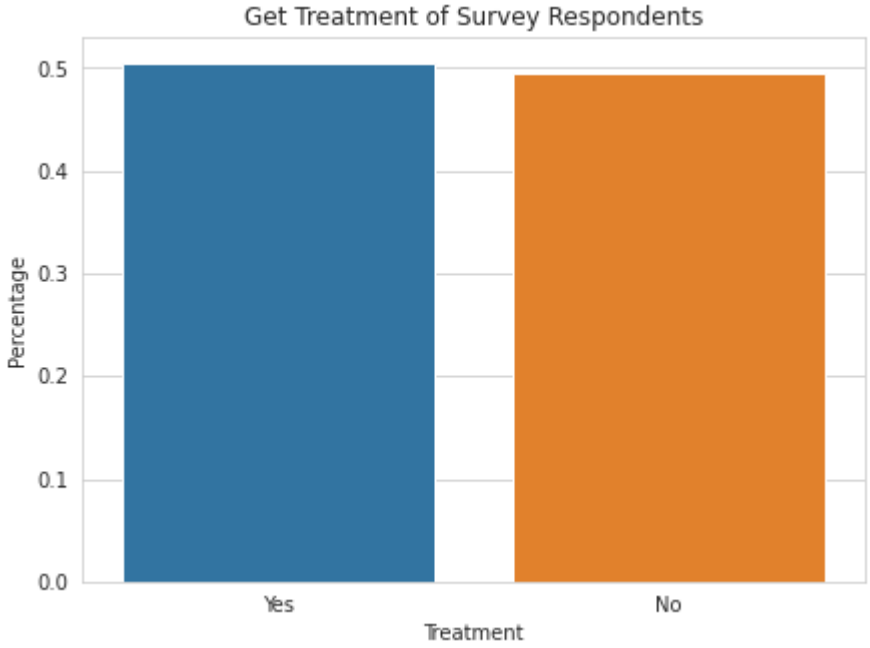
In [ ]:

In [ ]:
```
EDA
```

In [ ]:

In [77]:
```
mh_eda = mh.copy()
```

In [78]:
```
sns.set_style("whitegrid")
plt.figure(figsize = (7,5))
eda_percentage = mh_eda['Treatment'].value_counts(normalize = True).rename_axis('Treatment').reset_index(name = 'Percen
sns.barplot(x = 'Treatment', y = 'Percentage', data = eda_percentage.head(10))
plt.title('Get Treatment of Survey Respondents')
plt.show()
```



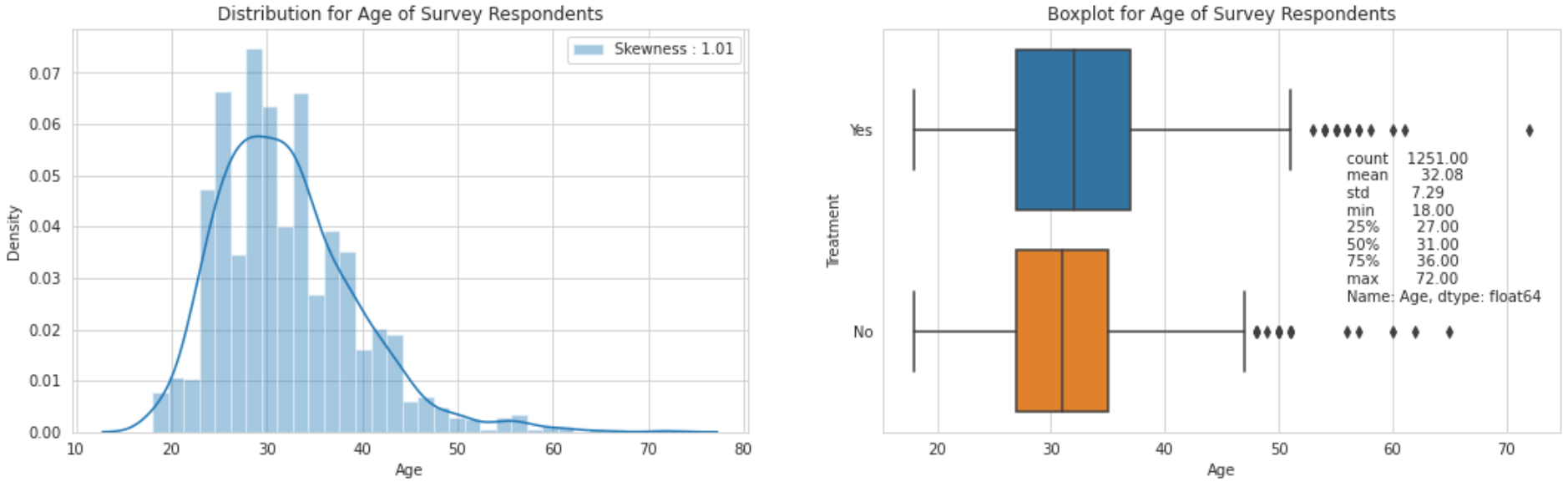
In [ ]:

In [ ]:
```
This is the respondents result of question,
'Have you get treatment for a mental health condition?'.
The percentage of respondents who want to get treatment is 50%. Workplaces
that promote mental health and support people with mental disorders are more
likely to reduce absenteeism, increase productivity and benefit from associated
economic gains. If employees enjoy good mental health, employees can:
make the most of your potential,
```

```
cope with what life throws at you,
play a full part in your relationships, your workplace, and your community.
I decided to separate them into 3 aspects to see what factors that company give
so employees want to get a treatment:
Employee's profiling
Employee's work environtment
Employee's mental health facilities
```

In [ ]:

In [79]:
```
plt.figure(figsize = (18,5))
plt.subplot(1,2,1)
sns.distplot(mh_eda['Age'], label = 'Skewness : %.2f'%(mh_eda['Age'].skew()))
plt.legend(loc = 0, fontsize = 10)
plt.title('Distribution for Age of Survey Respondents')
plt.subplot(1,2,2)
sns.boxplot(x = "Age", y = "Treatment", data = mh_eda)
plt.title('Boxplot for Age of Survey Respondents')
age = str(mh_eda['Age'].describe().round(2))
plt.text(56, 0.85, age)
plt.show()
```

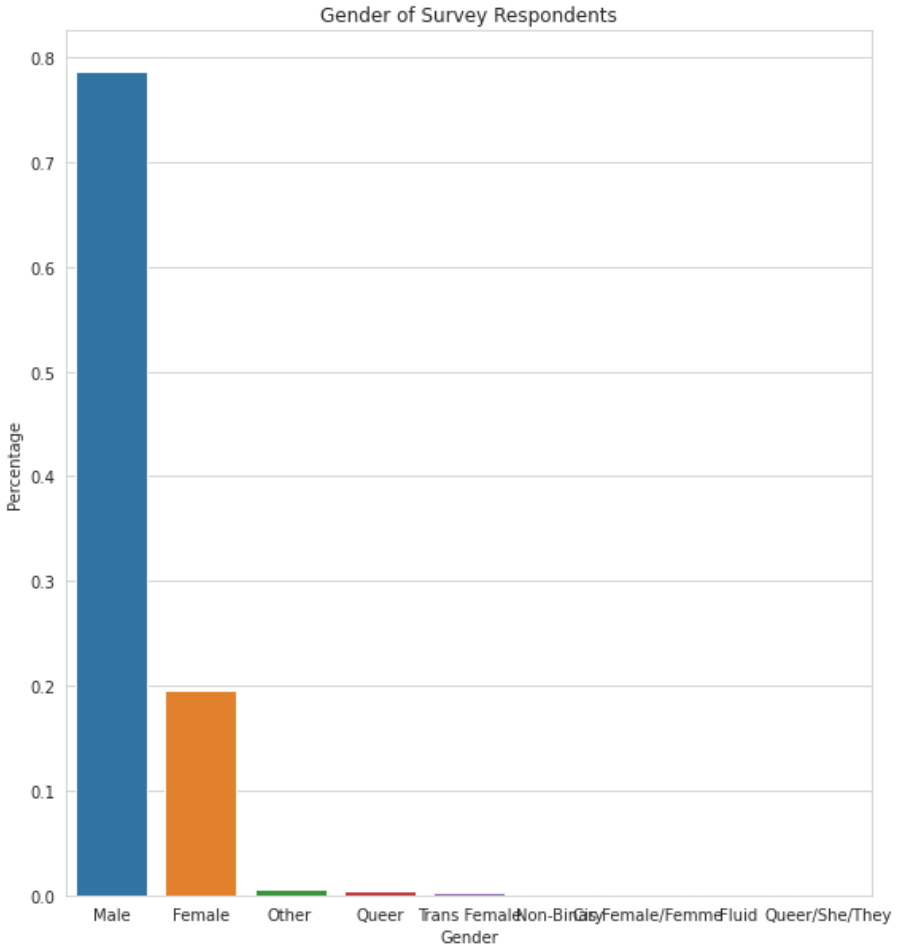


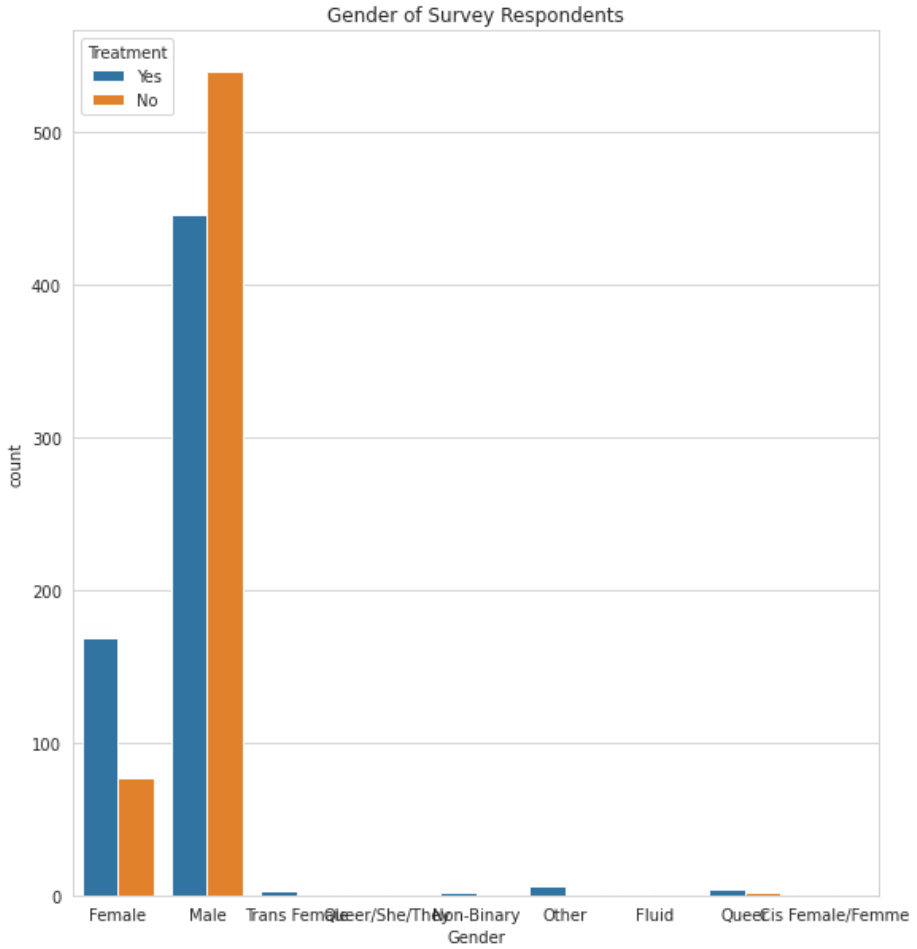
In [ ]:

In [ ]:
```
Skewness
Based on the plot, the skewness score is 1.01, which means the data are highly
skewed and with Positive skewness where the mode is
smaller than mean or median.
It's indicated that most of the employees that fill the survey around the end
20s to early 40s. I assume that they on between
mid to senior-level positions. The distribution of ages is right-skewed which
is expected as the tech industry tends to have younger employees.
From an article that I read, young (usually white, mostly male) faces of start-up
founders like Mark
Zuckerberg and other “tech bros” have become the symbol and stereotypical image that tends
to represent the tech industry.
Boxplot
From the boxplot, there is no statistically significant difference of ages between respondents
that get treatment and no treatment.
```

In [ ]:
```
```

In [81]:
```
plt.figure(figsize = (20,10))
plt.subplot(1,2,1)
eda_percentage = mh_eda['Gender'].value_counts(normalize = True).rename_axis('Gender').reset_index(name = 'Percentage')
sns.barplot(x = 'Gender', y = 'Percentage', data = eda_percentage.head(10))
plt.title('Gender of Survey Respondents')
plt.subplot(1,2,2)
sns.countplot(mh_eda['Gender'], hue = mh_eda['Treatment'])
plt.title('Gender of Survey Respondents')
plt.show()
```





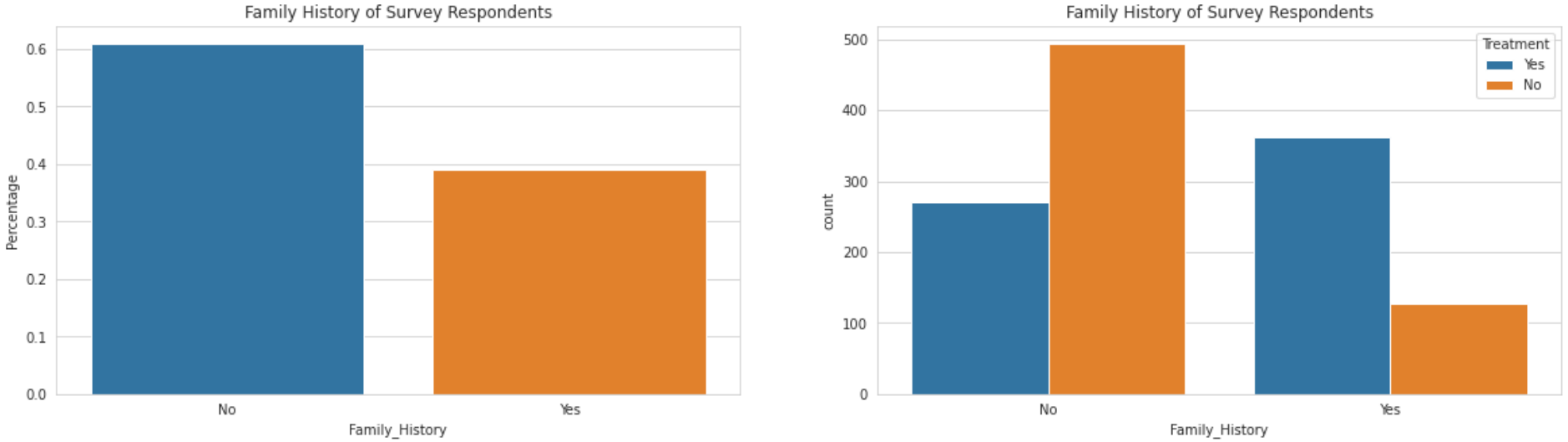
In [ ]:

In [ ]:
```
This is the respondents result of question, 'What is your gender identities?'.
Almost 79% of respondents are male, not surprisingly, especially in the tech field.
The very large gap between men and women causes
higher competitive pressure for women than men. Based on the plot, female that want
to get treatment is high around 70%. Maybe some
of them get sexual harrassment or racism at work because female are scarce in the tech industry.
There is a Queer entry of less than 2%. Although the percentage of queer is very low,
it still deserves to dig out some new
insights. For example, such a small proportion can show a significant difference in
the count of who wants the treatments,
indicating that for the queer, mental health problems are serious too.
```

```
In my opinion, maybe they received hate speech or
discrimination in the workplace.
```

In [ ]:

In [82]:
```
plt.figure(figsize = (20,5))
plt.subplot(1,2,1)
eda_percentage = mh_eda['Family_History'].value_counts(normalize = True).rename_axis('Family_History').reset_index(name
sns.barplot(x = 'Family_History', y = 'Percentage', data = eda_percentage)
plt.title('Family History of Survey Respondents')
plt.subplot(1,2,2)
sns.countplot(mh_eda['Family_History'], hue = mh_eda['Treatment'])
plt.title('Family History of Survey Respondents')
plt.show()
```



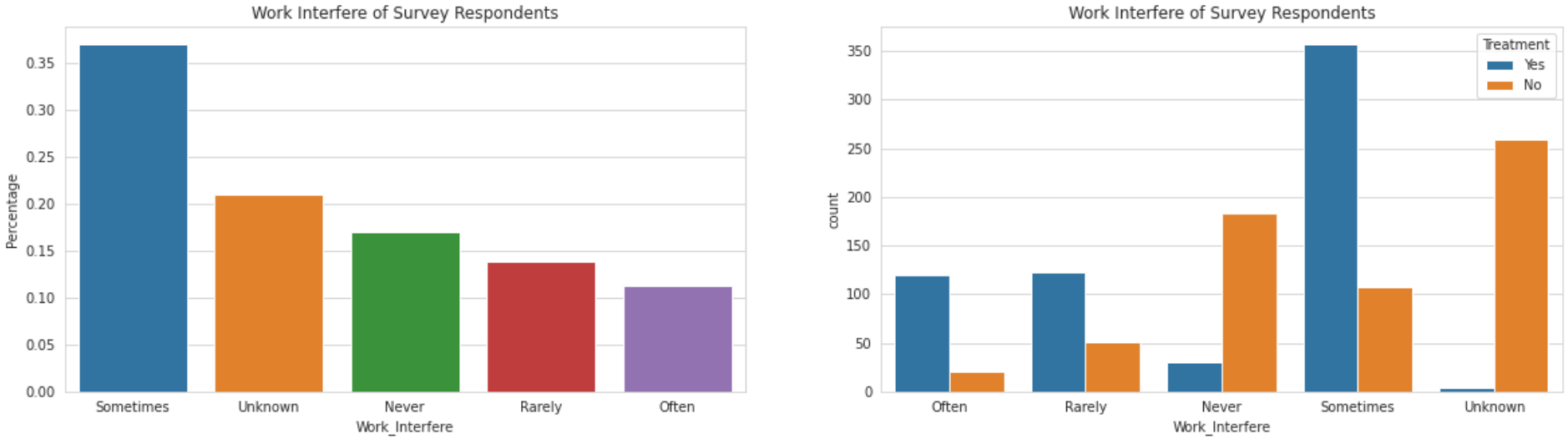
In [ ]:

In [ ]:
```
This is the respondents result of question,
'Do you have a family history of mental illness?'.
From 40% of respondents who say that they have a family history
of mental illness, the plot shows that they significantly
want to get treatment rather than without a family history. This
is acceptable, remember the fact that people with a family
history pay more attention to mental illness. Family history is a
significant risk factor for many mental health disorders.
```

```
The apple does not fall far from the tree, as it is relatively
common for families with mental illness symptoms to have one
or more relatives with histories of similar difficulties.
```

In [ ]:

In [83]:
```
plt.figure(figsize = (20,5))
plt.subplot(1,2,1)
eda_percentage = mh_eda['Work_Interfere'].value_counts(normalize = True).rename_axis('Work_Interfere').reset_index(name
sns.barplot(x = 'Work_Interfere', y = 'Percentage', data = eda_percentage)
plt.title('Work Interfere of Survey Respondents')
plt.subplot(1,2,2)
sns.countplot(mh_eda['Work_Interfere'], hue = mh_eda['Treatment'])
plt.title('Work Interfere of Survey Respondents')
plt.show()
```



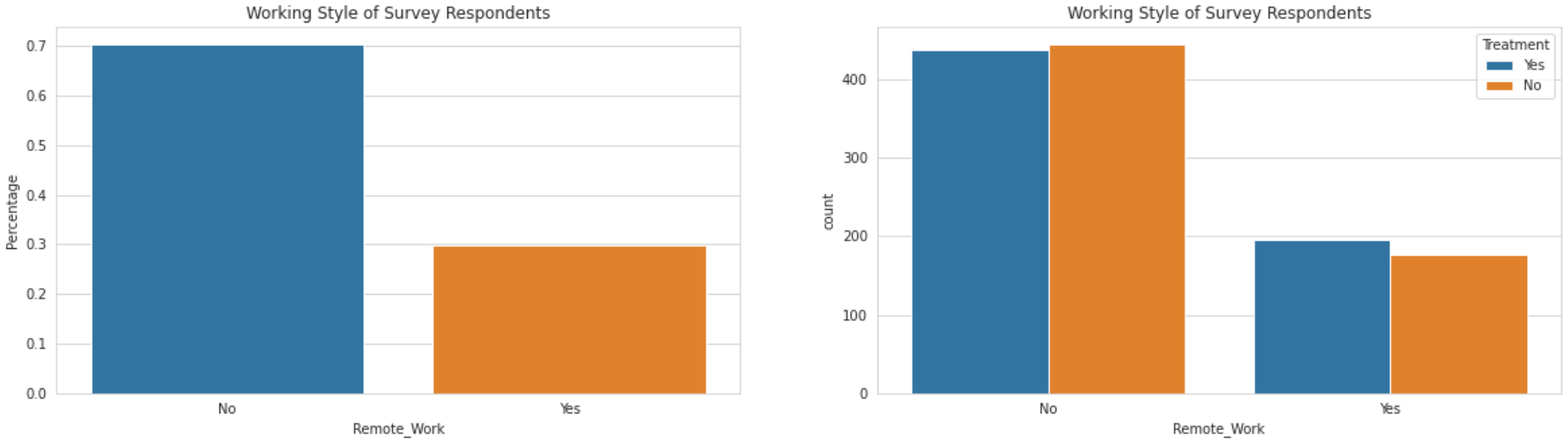
In [ ]:

In [ ]:
```
This is the respondents result of question, 'If you have a mental
health condition, do you feel that it interferes with your work?'.
About 78% of respondents have experienced interference at work with
a ratio of rarely, sometimes, and frequently.
Mental health conditions sometimes become an interfere while working
about 45%. The plots prove that almost 80%
want to get treatment. But it's surprising to know even mental health
```

```
never has interfered at work, there is a little
group that still want to get treatment before it become a job stress.
It can be triggered by the requirements of the
job do not match the capabilities, resources or needs of the worker.
```

In [ ]:

In [84]:
```
plt.figure(figsize = (20,5))
plt.subplot(1,2,1)
eda_percentage = mh_eda['Remote_Work'].value_counts(normalize = True).rename_axis('Remote_Work').reset_index(name = 'Pe
sns.barplot(x = 'Remote_Work', y = 'Percentage', data = eda_percentage)
plt.title('Working Style of Survey Respondents')
plt.subplot(1,2,2)
sns.countplot(mh_eda['Remote_Work'], hue = mh_eda['Treatment'])
plt.title('Working Style of Survey Respondents')
plt.show()
```



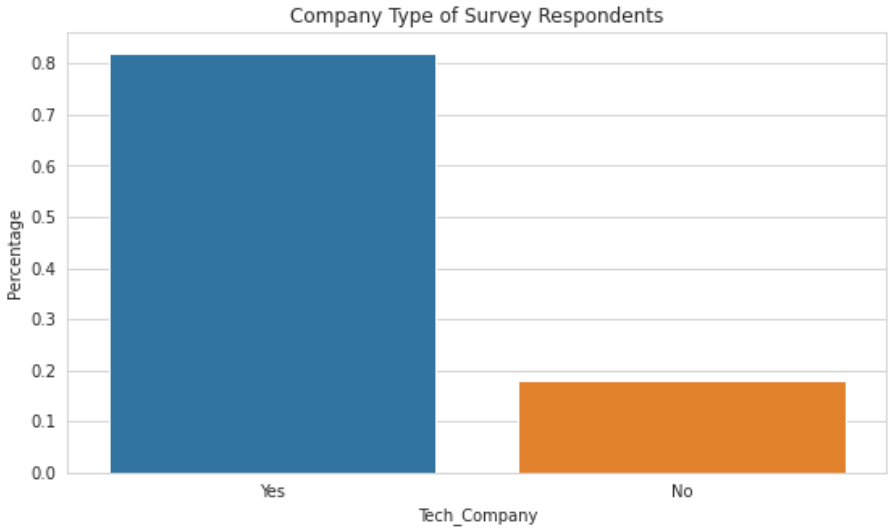
In [ ]:

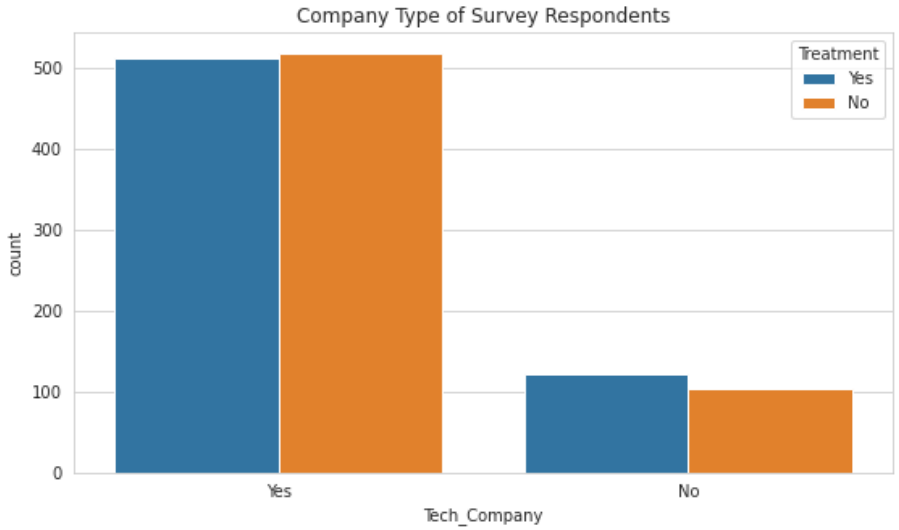
In [ ]:
```
This is the respondents result of question, Do you work remotely
(outside of an office) at least 50% of the time?.
Around 70% of respondents don't work remotely, which means the
biggest factor of mental health disorder came up triggered
on the workplace. On the other side, it has slightly different
between an employee that want to get treatment and don't want
```

```
to get a treatment. But it's getting interesting when we see a
respondent who works 50% of the workday remotely. The employee
who want to get treatment is a little bit higher. I have no idea
why those employees work remotely to analyze more because the data
doesn't provide that information.
```

In [ ]:

In [85]:
```
plt.figure(figsize = (20,5))
plt.subplot(1,2,1)
eda_percentage = mh_eda['Tech_Company'].value_counts(normalize = True).rename_axis('Tech_Company').reset_index(name = '
sns.barplot(x = 'Tech_Company', y = 'Percentage', data = eda_percentage)
plt.title('Company Type of Survey Respondents')
plt.subplot(1,2,2)
sns.countplot(mh_eda['Tech_Company'], hue = mh_eda['Treatment'])
plt.title('Company Type of Survey Respondents')
plt.show()
```





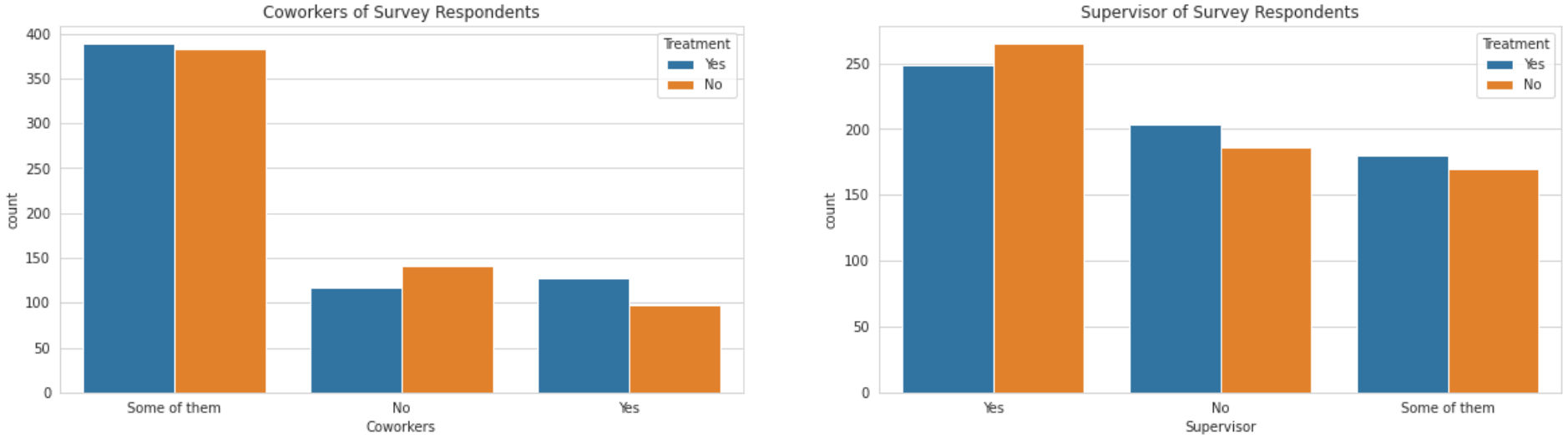
In [ ]:

In [ ]:
```
This is the respondents result of question, 'Is your employer
primarily a tech company/organization?'.
Even the main target of the survey is the tech field, there are
18% of companies belong to the non-tech field. But it
can be seen from the plot whether the company belongs to the
```

```
tech field or not, mental health still becomes a big problem.
I think the environment affects a lot of employees and some of
them can't take it for granted like abuse at the workplace.
However, I found that the number of employees in the technology
field that want to get treatment is slightly lower than no
treatment. But the non-technical field is the opposite. Maybe
the non-tech company give more support for employee to get treatment?
```

In [ ]:

In [86]:
```
plt.figure(figsize = (20,5))
plt.subplot(1,2,1)
sns.countplot(mh_eda['Coworkers'], hue = mh_eda['Treatment'])
plt.title('Coworkers of Survey Respondents')
plt.subplot(1,2,2)
sns.countplot(mh_eda['Supervisor'], hue = mh_eda['Treatment'])
plt.title('Supervisor of Survey Respondents')
plt.show()
```



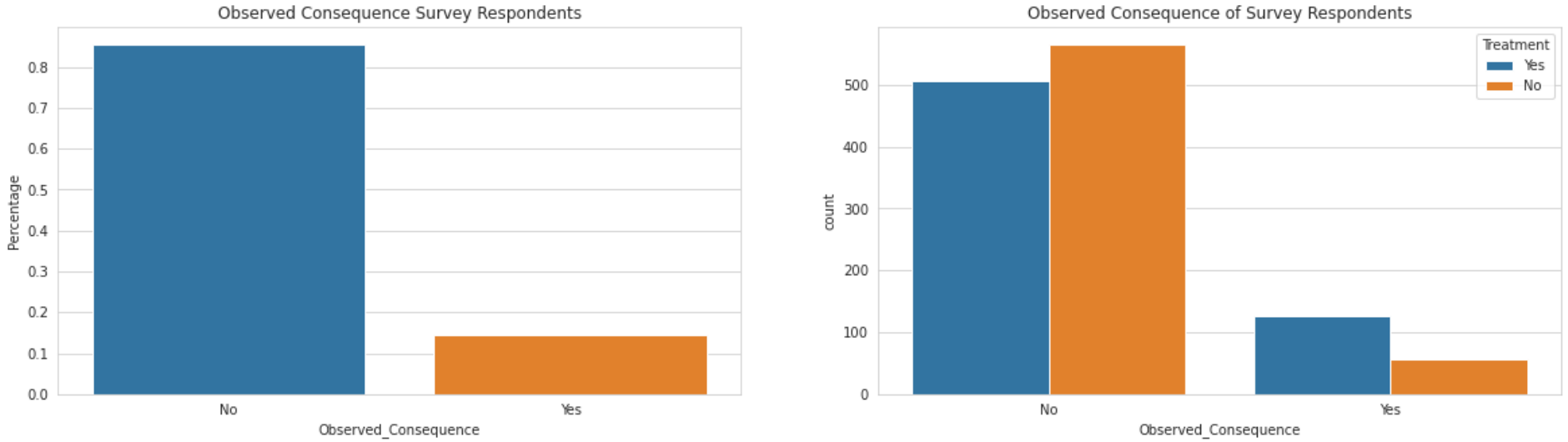
In [ ]:

In [ ]:
```
This is the respondents result of question, 'Would you be willing to
discuss a mental health issue with your coworkers?'.
From 18% of respondents who say yes to discuss it with coworkers,
60% of them want to get treatment.
```

```
About 60% of respondents decide to discuss some of them with coworkers.
Employees who do that and want to get treatment are half
of them. Let's see if the respondent will discuss it with a supervisor or not.
This is the respondents result of question, 'Would you be willing to
discuss a mental health issue with your direct supervisor(s)?'.
From 40% of respondents who say yes to discuss with supervisor, only
55% of them that want to get treatment. I think maybe talking
to someone in a higher position could help the relief.
It's the opposite while employees discuss with coworkers.
```

In [ ]:

In [87]:
```
plt.figure(figsize = (20,5))
plt.subplot(1,2,1)
eda_percentage = mh_eda['Observed_Consequence'].value_counts(normalize = True).rename_axis('Observed_Consequence').rese
sns.barplot(x = 'Observed_Consequence', y = 'Percentage', data = eda_percentage)
plt.title('Observed Consequence Survey Respondents')
plt.subplot(1,2,2)
sns.countplot(mh_eda['Observed_Consequence'], hue = mh_eda['Treatment'])
plt.title('Observed Consequence of Survey Respondents')
plt.show()
```

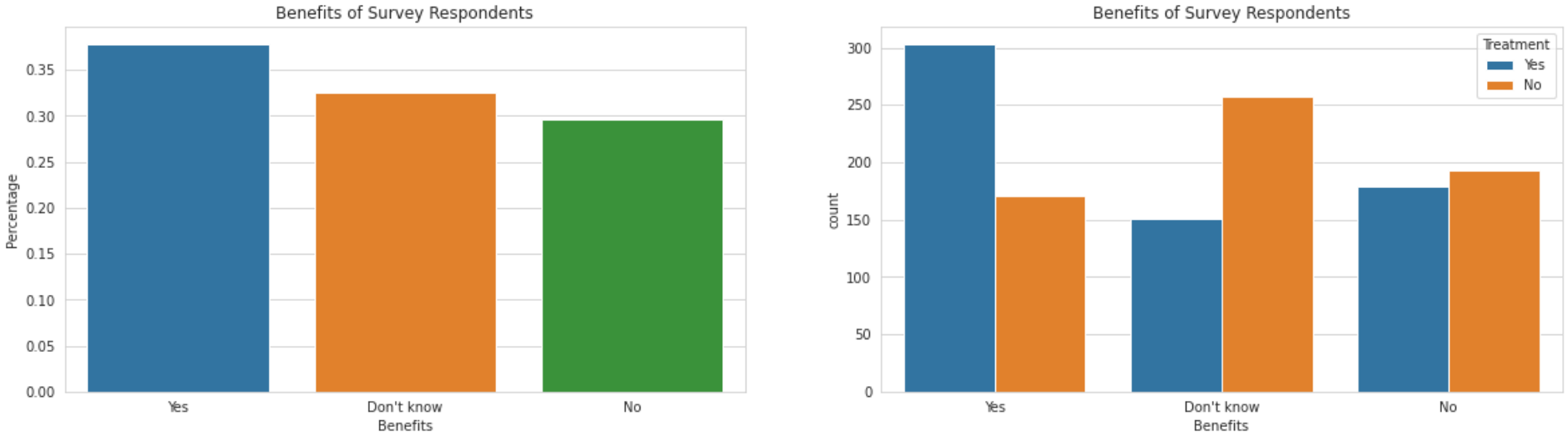


In [ ]:

In [ ]:
```
This is the respondents result of question,
'Have you heard of or observed negative consequences for coworkers with mental health
conditions in your workplace?'.
From 15% of respondents who say yes about knowing the
negative consequences for coworkers with mental heatlh condition,
almost 70% of them that want to get treatment. After the employee
knows about the negative consequences, it becomes a good
trigger for someone to get treatment to prevent mental health conditions.
```

In [ ]:
```
```

In [88]:
```
plt.figure(figsize = (20,5))
plt.subplot(1,2,1)
eda_percentage = mh_eda['Benefits'].value_counts(normalize = True).rename_axis('Benefits').reset_index(name = 'Percenta
sns.barplot(x = 'Benefits', y = 'Percentage', data = eda_percentage)
plt.title('Benefits of Survey Respondents')
plt.subplot(1,2,2)
sns.countplot(mh_eda['Benefits'], hue = mh_eda['Treatment'])
plt.title('Benefits of Survey Respondents')
plt.show()
```

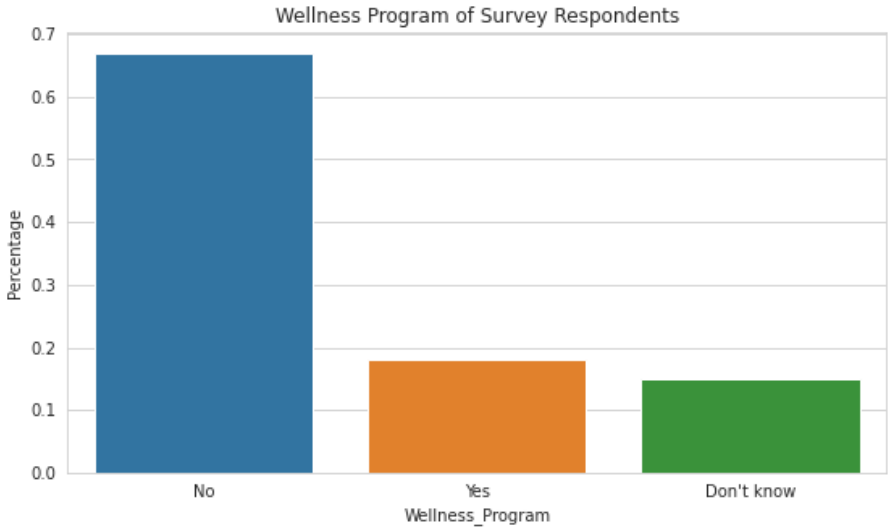


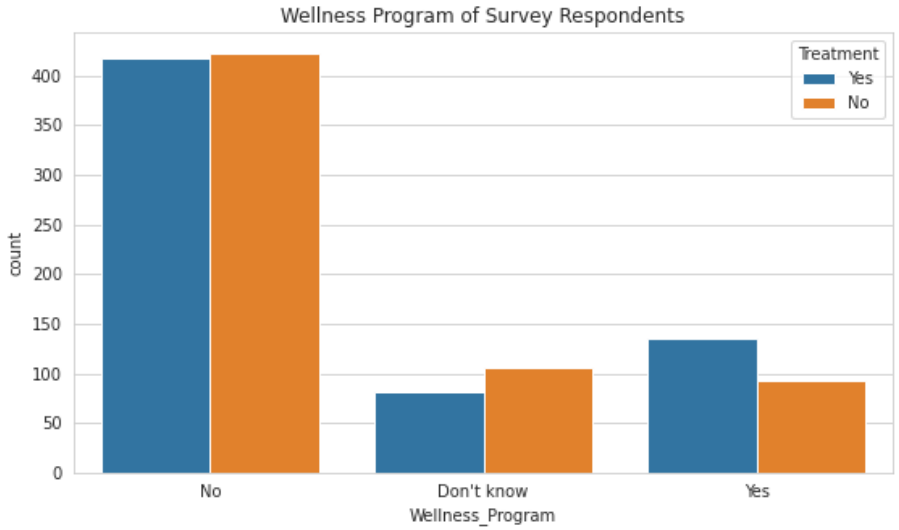
In [ ]:
```
```

In [ ]:
```
This is the respondents result of question, 'Does your employer
mental health benefits?'.
Only 35% of respondents know about mental health benefits that the
company provides for them.
For employees who know the benefits, almost 60% of the employees want to get treatment.
Surprisingly, there is an employee who
doesn't know and says that the company doesn't provide still want to get treatment.
I assume that maybe the company can't
provide it properly because of budgeting or financial struggling.
```

In [ ]:
```
```

In [89]:
```
plt.figure(figsize = (20,5))
plt.subplot(1,2,1)
eda_percentage = mh_eda['Wellness_Program'].value_counts(normalize = True).rename_axis('Wellness_Program').reset_index(
sns.barplot(x = 'Wellness_Program', y = 'Percentage', data = eda_percentage)
plt.title('Wellness Program of Survey Respondents')
plt.subplot(1,2,2)
sns.countplot(mh_eda['Wellness_Program'], hue = mh_eda['Treatment'])
plt.title('Wellness Program of Survey Respondents')
plt.show()
```



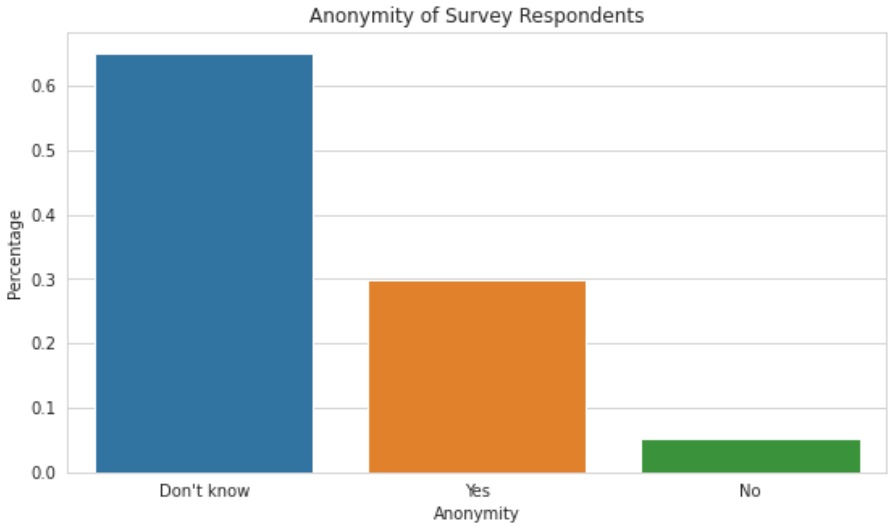


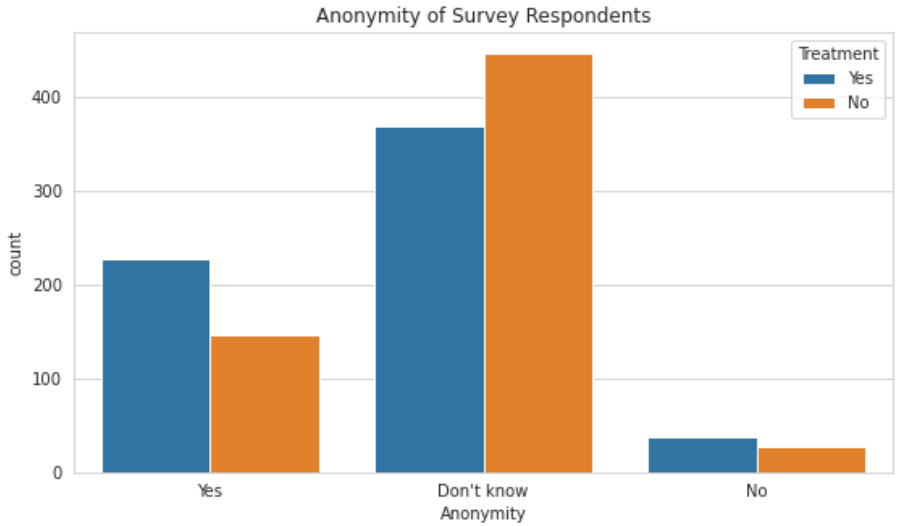
In [ ]:
```
```

In [ ]:
```
This is the respondents result of question,
'Has your employer ever discussed mental health as part of an employee wellness program?'.
About 19% of the repondents say yes about become a part of employee wellness program
and 60% of employee want to get treatment. After
become a part of wellness program, i assume that employee feels a good vibe about it.
More than 65% of respondents say that there aren't any wellness programs that provide
by their company. But half of the respondents want to get treatment, which means the company need to provide it soon.
Based on my curiosity about company's benefit before, I think it makes sense if
it's about company budgeting. I know it will spend a lot of money, moreover, the
company has a lot of employees to taking care of. My second thought, it's still about
budgeting but for a small company, it will be a lot of struggle.
```

In [ ]:
```
```

In [90]:
```
plt.figure(figsize = (20,5))
plt.subplot(1,2,1)
eda_percentage = mh_eda['Anonymity'].value_counts(normalize = True).rename_axis('Anonymity').reset_index(name = 'Percen
sns.barplot(x = 'Anonymity', y = 'Percentage', data = eda_percentage)
plt.title('Anonymity of Survey Respondents')
plt.subplot(1,2,2)
sns.countplot(mh_eda['Anonymity'], hue = mh_eda['Treatment'])
plt.title('Anonymity of Survey Respondents')
plt.show()
```





In [ ]:

In [ ]:
```
This is the respondents result of question,
'Is your anonymity protected if you choose to take advantage of mental health or
substance abuse treatment resources?'.
About 30% of respondents say yes if their anonymity is protected while taking advantage
of mental health or substance abuse treatment resources and almost 65% of employees want
to get treatment. The employee feels that the company protected their privacy and it's a
good move for the company to build trust with their employees. Because of that, the employee wants to get
treatment to be better.
```

In [ ]:

In [ ]:

In [ ]:
```
Conclusion

Nearly 86% of employees report improved work performance and
lower rates of absenteeism after receiving treatment for depression,
according to an April 2018 article in the Journal of Occupational
and Environmental Medicine. This means big gains in retention
and productivity for employers. By providing employees access to mental
health benefits, the company can begin to create a culture of
understanding and compassion at the tech company. And having employees
```

```
who feel cared for and happy isn't just good, it's good business.

Based on profiling the respondents
Companies must know that gender and family history greatly influence the
decision to get treatment for employees. So if the company wants
to provide more support, the company must make an assessment of the
employee's personality because different characters can determine
different needs. Age can also be a trigger, considering that most of
them are young so there is a high chance that they will be
open-minded to get treatment.

Based on the work environment of respondents
Work interference is the most influential of employees who want to get
treatment. This means the company should consider
providing facilities to anticipate job stress on employees. Some of
the companies decide to make a private room or silent room in
case employees suddenly feel stress and need a private moment to relieve.

Based on the mental health facilities of respondents
The company needs to provide a good benefit for employees so they can
maintain their mental health. If the company can don't
have resources for it, there are so many third parties who can be
hired to maintain a wellness program for the company.
Building trust like keep private about whom employee that gets treatment
also can also give a trigger for employee want to get treatment.
```

In [ ]:

In [ ]:
```
PreProcessing

Preprocessing Scheme

OneHotEncoding: Gender, Family History, Employee Numbers, Remote
    Work, Tech Company, Benefits, Care Options, Wellness Program,
    Seek Help, Anonymity, Medical Leave, Mental Health Consequence,
    Physical Health Consequence, Coworkers, Supervisor,
    Mental Health Interview, Physical Health Interview,
    Mental VS Physical, Observed Consequence
Simple Imputer Most Frequent: Self Employed, Work Interfere
Iterative Impute: Age
Target: Treatment
```

In [ ]:

In [91]:
```
mode_onehot_pipe = Pipeline([
    ('encoder', SimpleImputer(strategy = 'most_frequent')),
    ('one hot encoder', OneHotEncoder(handle_unknown = 'ignore'))])

transformer = ColumnTransformer([
    ('one hot', OneHotEncoder(handle_unknown = 'ignore'), ['Gender', 'Family_History', 'Employee_Numbers',
                                                           'Remote_Work', 'Tech_Company', 'Benefits', 'Care_Options',
                                                           'Wellness_Program', 'Seek_Help', 'Anonymity',
                                                           'Medical_Leave', 'Mental_Health_Consequence',
                                                           'Physical_Health_Consequence', 'Coworkers', 'Supervisor',
                                                           'Mental_Health_Interview', 'Physical_Health_Interview',
                                                           'Mental_VS_Physical', 'Observed_Consequence']),
    ('mode_onehot_pipe', mode_onehot_pipe, ['Self_Employed', 'Work_Interfere']),
    ('iterative', IterativeImputer(max_iter = 10, random_state = 0), ['Age'])])
```

In [92]:
```
mh['Treatment'].value_counts()/mh.shape[0]*100
```

Out[92]:
```
Yes    50.478469
No     49.521531
Name: Treatment, dtype: float64
```

In [93]:
```
mh['Treatment'] = np.where(mh['Treatment'] == 'Yes', 1, 0)
```

In [94]:
```
X = mh.drop('Treatment', axis = 1)
y = mh['Treatment']
```

In [95]:
```
X.shape
```

Out[95]:
```
(1254, 22)
```

In [96]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y,
                                                    stratify = y,
                                                     test_size = 0.3,
                                                    random_state = 2222)
```

In [ ]:

In [ ]:

In [ ]:
```
Modeling
Define Model

In supervised learning, algorithms learn from labeled data.
In this case, I use Classification technique for determining which
class is yes and no.
I use 3 basic models and 4 ensemble models to predict.
Basic models:
Logistic Regression (logreg)
Decision Tree Classifier (tree)
K-Nearest Neighbor (knn)
Ensemble models:
Random Forest Classifier (rf)
Ada Boost Classifier (ada)
Gradient Boosting Classifier (grad)
```

In [ ]:

In [ ]:

In [97]:
```
logreg = LogisticRegression()
tree = DecisionTreeClassifier(random_state = 2222)
knn = KNeighborsClassifier()
rf = RandomForestClassifier(random_state = 2222)
ada = AdaBoostClassifier(random_state = 2222)
grad = GradientBoostingClassifier(random_state = 2222)
```

In [ ]:

In [ ]:
```
Cross Validation:
```

In [ ]:

In [98]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
```

```
# PreProcessing
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.pipeline import Pipeline
from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import SimpleImputer, IterativeImputer
from sklearn.preprocessing import MinMaxScaler

# Splitting Data
from sklearn.model_selection import train_test_split, StratifiedKFold, cross_val_score

# Modeling
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, recall_score, plot_roc_curve
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier, GradientBoostingClassifier

logreg_pipe = Pipeline([('transformer', transformer), ('logreg', logreg)])
tree_pipe = Pipeline([('transformer', transformer), ('tree', tree)])
knn_pipe = Pipeline([('transformer', transformer), ('knn', knn)])
rf_pipe = Pipeline([('transformer', transformer), ('rf', rf)])
ada_pipe = Pipeline([('transformer', transformer), ('ada', ada)])
grad_pipe = Pipeline([('transformer', transformer), ('grad', grad)])

def model_evaluation(model, metric):
    model_cv = cross_val_score(model, X_train, y_train, cv=StratifiedKFold(n_splits=5), scoring=metric)
    return model_cv

logreg_pipe_cv = model_evaluation(logreg_pipe, 'recall')
tree_pipe_cv = model_evaluation(tree_pipe, 'recall')
knn_pipe_cv = model_evaluation(knn_pipe, 'recall')
rf_pipe_cv = model_evaluation(rf_pipe, 'recall')
ada_pipe_cv = model_evaluation(ada_pipe, 'recall')
grad_pipe_cv = model_evaluation(grad_pipe, 'recall')

for model in [logreg_pipe, tree_pipe, knn_pipe, rf_pipe, ada_pipe, grad_pipe]:
    model.fit(X_train, y_train)

score_cv = [logreg_pipe_cv.round(5), tree_pipe_cv.round(5), knn_pipe_cv.round(5),
            rf_pipe_cv.round(5), ada_pipe_cv.round(5), grad_pipe_cv.round(5)]
score_mean = [logreg_pipe_cv.mean(), tree_pipe_cv.mean(), knn_pipe_cv.mean(), rf_pipe_cv.mean(),
              ada_pipe_cv.mean(), grad_pipe_cv.mean()]
```

```
score_std = [logreg_pipe_cv.std(), tree_pipe_cv.std(), knn_pipe_cv.std(), rf_pipe_cv.std(),
             ada_pipe_cv.std(), grad_pipe_cv.std()]
score_recall_score = [recall_score(y_test, logreg_pipe.predict(X_test)),
            recall_score(y_test, tree_pipe.predict(X_test)),
            recall_score(y_test, knn_pipe.predict(X_test)),
            recall_score(y_test, rf_pipe.predict(X_test)),
            recall_score(y_test, ada_pipe.predict(X_test)),
            recall_score(y_test, grad_pipe.predict(X_test))
           ]

method_name = ['Logistic Regression', 'Decision Tree Classifier', 'KNN Classifier', 'Random Forest Classifier',
               'Ada Boost Classifier', 'Gradient Boosting Classifier']

cv_summary = pd.DataFrame({
    'method': method_name,
    'cv score': score_cv,
    'mean score': score_mean,
    'std score': score_std,
    'recall score': score_recall_score
})

cv_summary
```

Out[98]:

	method	cv score	mean score	std score	recall score
0	Logistic Regression	[0.80899, 0.88764, 0.88764, 0.88636, 0.875]	0.869127	0.030442	0.863158
1	Decision Tree Classifier	[0.76404, 0.68539, 0.83146, 0.84091, 0.79545]	0.783453	0.056111	0.747368
2	KNN Classifier	[0.57303, 0.67416, 0.70787, 0.65909, 0.65909]	0.654648	0.044527	0.605263
3	Random Forest Classifier	[0.77528, 0.83146, 0.88764, 0.90909, 0.86364]	0.853422	0.046824	0.810526
4	Ada Boost Classifier	[0.80899, 0.85393, 0.92135, 0.84091, 0.85227]	0.855490	0.036674	0.857895
5	Gradient Boosting Classifier	[0.78652, 0.80899, 0.89888, 0.89773, 0.86364]	0.851149	0.045953	0.800000

In [ ]:

In [ ]: HyperParam Tuning:

In [ ]:

In [99]:
```
lr_estimator = Pipeline([
    ('transformer', transformer),
    ('model', logreg)])

hyperparam_space = {
    'model__C': [ 1, 0.5, 0.1, 0.05, 0.01],
    'model__solver': ['newton-cg', 'lbfgs', 'liblinear'],
    'model__class_weight': ['balanced', 'dict'],
    'model__max_iter': [100, 200, 300],
    'model__multi_class': ['auto', 'ovr', 'multinomial'],
    'model__random_state': [2222]
}

grid_lr = GridSearchCV(
                lr_estimator,
                param_grid = hyperparam_space,
                cv = StratifiedKFold(n_splits = 5),
                scoring = 'recall',
                n_jobs = -1)

grid_lr.fit(X_train, y_train)

print('best score', grid_lr.best_score_)
print('best param', grid_lr.best_params_)
```

```
best score 0.8781664964249234
best param {'model__C': 0.1, 'model__class_weight': 'balanced', 'model__max_iter': 100, 'model__multi_class': 'multinom
ial', 'model__random_state': 2222, 'model__solver': 'newton-cg'}
```

In [100…
```
logreg_pipe.fit(X_train, y_train)
recall_logreg = (recall_score(y_test, logreg_pipe.predict(X_test)))

grid_lr.best_estimator_.fit(X_train, y_train)
recall_grid = (recall_score(y_test, grid_lr.predict(X_test)))

score_list = [recall_logreg, recall_grid]
method_name = ['Logistic Regression Before Tuning', 'Logistic Regression After Tuning']
best_summary = pd.DataFrame({
    'method': method_name,
    'score': score_list
})
best_summary
```

Out[100…

	method	score
0	Logistic Regression Before Tuning	0.863158
1	Logistic Regression After Tuning	0.826316

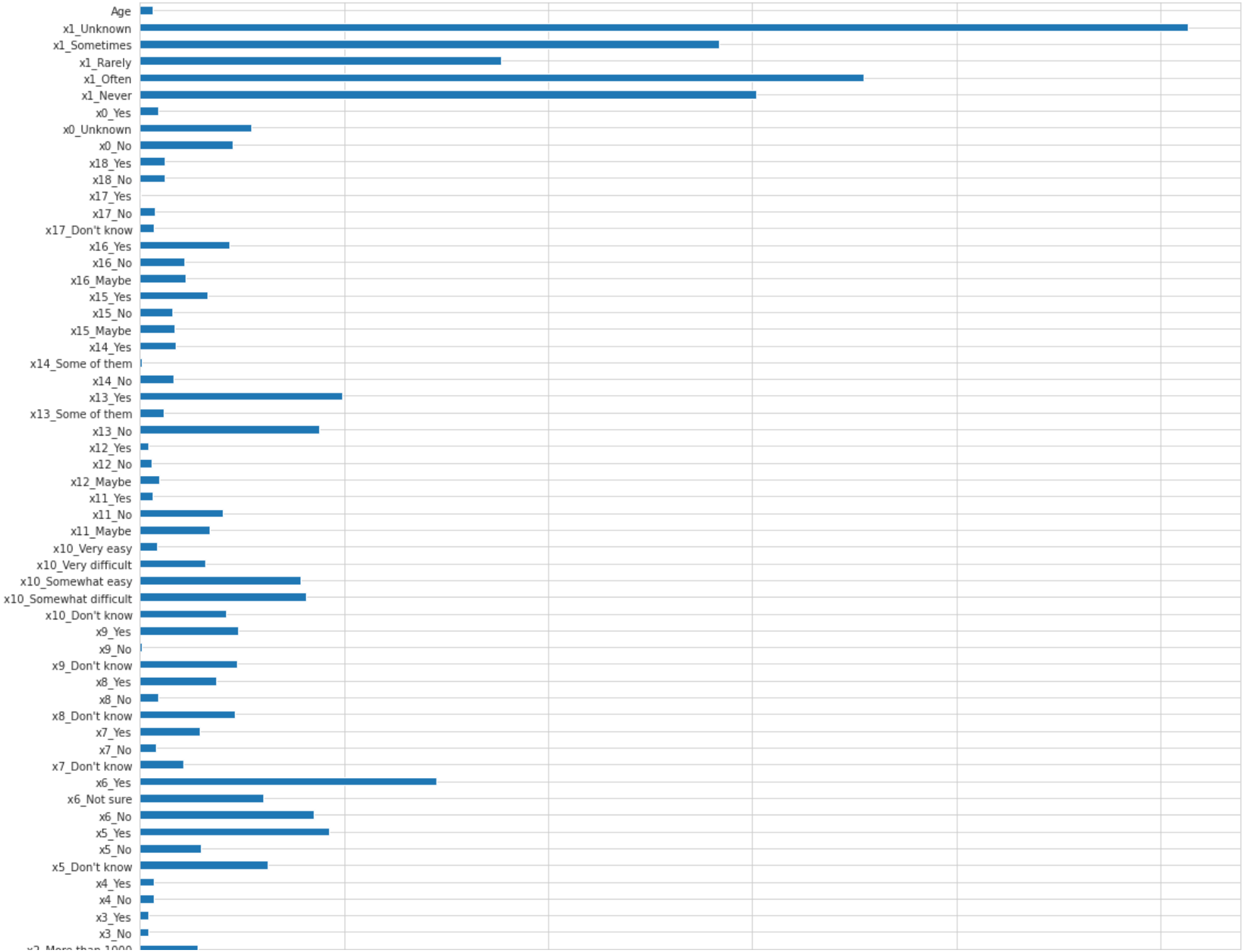
In [ ]:

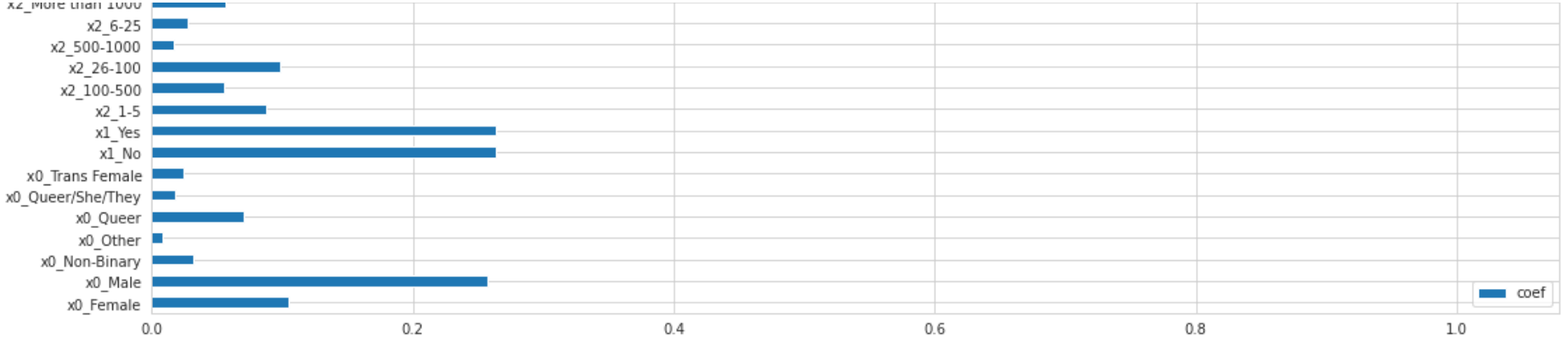
In [ ]:
```
Feature Selection using Tuning Result:
```

In [ ]:

In [101…
```
features = list(transformer.transformers_[0][1].get_feature_names())+list(transformer.transformers_[1][1][1].get_featur
coef_table = pd.DataFrame({'coef': grid_lr.best_estimator_[1].coef_.flatten()}, index = features)
abs(coef_table).plot(kind = 'barh', figsize = (18,20))
```

Out[101…
```
<AxesSubplot:>
```


Age
x1_Unknown
x1_Sometimes
x1_Rarely
x1_Often
x1_Never
x0_Yes
x0_Unknown
x0_No
x18_Yes
x18_No
x17_Yes
x17_No
x17_Don't know
x16_Yes
x16_No
x16_Maybe
x15_Yes
x15_No
x15_Maybe
x14_Yes
x14_Some of them
x14_No
x13_Yes
x13_Some of them
x13_No
x12_Yes
x12_No
x12_Maybe
x11_Yes
x11_No
x11_Maybe
x10_Very easy
x10_Very difficult
x10_Somewhat easy
x10_Somewhat difficult
x10_Don't know
x9_Yes
x9_No
x9_Don't know
x8_Yes
x8_No
x8_Don't know
x7_Yes
x7_No
x7_Don't know
x6_Yes
x6_Not sure
x6_No
x5_Yes
x5_No
x5_Don't know
x4_Yes
x4_No
x3_Yes
x3_No



In [ ]:

In [ ]:
```
Based on selecting features based on coefficient score,
I decided to drop 4 features manually who gets a score under 0.05 for
all answer choices for every feature. There are Age, x3(Remote_work),
x7(Wellness_Program), x12(Physical_Health_Consequence).
```

In [ ]:

In [ ]:

In [ ]:
```
Re-run Using Feature Selection
linkcode
Preprocessing Scheme

OneHotEncoding: Gender, Family History, Employee Numbers, Tech Company,
    Benefits, Care Options, Seek Help, Anonymity, Medical Leave, Mental Health
    Consequence, Coworkers, Supervisor, Mental Health Interview, Physical Health Interview, Mental_VS_Physical,
Observed_Consequence
Mode: Self Employed, Work Interfere
Target: Treatment
```

In [ ]:

In [102…
```
mh_tuning = mh.copy()
mh_tuning.drop(columns = ['Age', 'Remote_Work', 'Wellness_Program', 'Physical_Health_Consequence'], inplace = True)
mh_tuning.head()
```

Out[102…

	Gender	Self_Employed	Family_History	Treatment	Work_Interfere	Employee_Numbers	Tech_Company	Benefits	Care_Options	Seek_Help	A
0	Female	Unknown	No	1	Often	6-25	Yes	Yes	Not sure	Yes	
1	Male	Unknown	No	0	Rarely	More than 1000	No	Don't know	No	Don't know	D
2	Male	Unknown	No	0	Rarely	6-25	Yes	No	No	No	D
3	Male	Unknown	Yes	1	Often	26-100	Yes	No	Yes	No	
4	Male	Unknown	No	0	Never	100-500	Yes	Yes	No	Don't know	D

In [103…

```
mode_onehot_pipe_second = Pipeline([
    ('encoder', SimpleImputer(strategy = 'most_frequent')),
    ('one hot encoder', OneHotEncoder(handle_unknown = 'ignore'))])

transformer_second = ColumnTransformer([
    ('one hot', OneHotEncoder(handle_unknown = 'ignore'), ['Gender', 'Family_History', 'Employee_Numbers',
                                                           'Tech_Company', 'Benefits', 'Care_Options',
                                                           'Seek_Help', 'Anonymity', 'Medical_Leave',
                                                           'Mental_Health_Consequence', 'Coworkers',
                                                           'Supervisor','Mental_Health_Interview',
                                                           'Physical_Health_Interview', 'Mental_VS_Physical',
                                                           'Observed_Consequence',]),
    ('mode_onehot_pipe', mode_onehot_pipe_second, ['Self_Employed', 'Work_Interfere']),])
```

In [104…

```
X_select = mh_tuning.drop('Treatment', axis = 1)
y_select = mh_tuning['Treatment']
```

In [105…

```
X_select_train, X_select_test, y_select_train, y_select_test = train_test_split(X_select,y_select,
                                                    stratify = y_select,
                                                     test_size = 0.3,
                                                    random_state = 2222)
```

In [107…

```
logreg_second = LogisticRegression(C = 0.5, class_weight = 'dict', max_iter = 100,
                                   multi_class = 'auto', random_state = 2222, solver = 'newton-cg')
logreg_second_pipe = Pipeline([('transformer', transformer_second), ('model', logreg_second)])
```

```
logreg_second_pipe.fit(X_select_train, y_select_train)
print('After Feature Selection Process, the score is ', recall_score(y_select_test, logreg_second_pipe.predict(X_select
```

```
After Feature Selection Process, the score is  0.868421052631579
```

In [ ]:

The insights from the analysis can measure campaign effectiveness and guide future strategies.

The results obtained from your thorough analysis of the dataset related to the mental health awareness campaign hold a wealth of valuable information that can be instrumental in gauging campaign effectiveness and charting the course for future strategies within the tech workplace. Let's delve into how these insights can be harnessed to fulfill these objectives:

1. Grasping the Present State:
- Begin by examining the current mindset and behaviors of professionals in the tech industry regarding mental health. This encompasses understanding the prevalence of mental health disorders, the existing stigma, and the readiness to seek help.

2. Benchmarking and Comparative Analysis:
- To measure the long-term impact of the campaign, compare the 2014 data with more recent data, if available. This comparison can reveal shifts in mental health awareness and attitudes over time.

3. Identification of Vital Metrics:
- Establish specific metrics that serve as yardsticks for campaign effectiveness. These could include tracking changes in the proportion of employees willing to openly discuss mental health in the workplace, those seeking assistance, or any reduction in the stigma associated with mental health issues.

4. Segmentation and Subgroup Analysis:
- Segment the dataset according to factors such as job roles, age, gender, and company size. This segmentation helps pinpoint subgroups in the tech workplace with differing attitudes and needs. For instance, it might unveil that younger employees are more open to mental health initiatives.

5. Correlation Investigations:
- Dive into correlations between various variables, like attitudes towards mental health and job satisfaction, productivity, or absenteeism. This exploration can shed light on direct or indirect effects of the campaign on workplace outcomes.

6. Highlighting Success Stories:
- Seek out instances where the campaign has had a notably positive impact, either on individuals or within specific companies. These success stories can serve as compelling case studies, offering evidence of the campaign's effectiveness.

7. Leveraging Feedback and Surveys:
- Whenever possible, conduct follow-up surveys and gather feedback from employees to understand their perceptions of the campaign. This qualitative data can complement the quantitative analysis.

8. Long-Term Analysis:
- If subsequent years' data is accessible, engage in longitudinal analysis to track trends over time. This can help determine whether the campaign's effects are sustained or diminishing.

9. Harnessing Predictive Modeling:
- Utilize predictive modeling to forecast the potential impact of future campaigns. Drawing insights from historical data, simulate different scenarios to assess the likely outcomes of diverse strategies.

10. Informed Recommendations:
- Building on the analysis, formulate a set of recommendations for future campaign strategies. These recommendations may encompass targeting specific demographics, focusing on destigmatization, or implementing educational initiatives.

11. Emphasizing Continuous Monitoring:
- Stress the significance of ongoing monitoring and iterative strategies. The insights extracted from your analysis should function as a launchpad for shaping and fine-tuning future campaigns.

12. Collaboration and Knowledge Sharing:
- Share your findings and collaborate with relevant stakeholders, including mental health organizations, HR departments, and company leadership. This collaborative approach is crucial in implementing and enhancing mental health initiatives.

Conclusion:

In a nutshell, the insights gleaned from our analysis will not only serve as a yardstick for evaluating the effectiveness of the mental health awareness campaign but also as a compass for shaping future strategies. The ultimate objective is to craft data-driven initiatives that have a lasting and positive impact on mental health awareness and support within the tech workplace.