data-analytics-phase-3-1

October 24, 2023

DATE:26/10/2023

TEAM ID-716

PROJECT NAME:Public health awareness campaign analysis using data analysis

IMPORT DEPENDENCIES

[2]:
```
import warnings
warnings.filterwarnings('ignore')
import numpy as np
import pandas as pd
import missingno as msno
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
%matplotlib inline
```

DATASET

[3]:
```
df = pd.read_csv("C:/Users/sjana/Downloads/survey.csv")
```

[4]:
```
df.head()
```

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

  family_history treatment work_interfere    no_employees  …  \
0             No       Yes          Often          Jun-25  …
1             No        No         Rarely  More than 1000  …
2             No        No         Rarely          Jun-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]
```

DATA EXPLORATION

```
[5]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1259 entries, 0 to 1258
Data columns (total 27 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Timestamp                  1259 non-null   object
 1   Age                        1259 non-null   int64
 2   Gender                     1259 non-null   object
 3   Country                    1259 non-null   object
 4   state                      744 non-null    object
 5   self_employed              1241 non-null   object
 6   family_history             1259 non-null   object
 7   treatment                  1259 non-null   object
 8   work_interfere             995 non-null    object
 9   no_employees               1259 non-null   object
 10  remote_work                1259 non-null   object
 11  tech_company               1259 non-null   object
 12  benefits                   1259 non-null   object
 13  care_options               1259 non-null   object
 14  wellness_program           1259 non-null   object
 15  seek_help                  1259 non-null   object
 16  anonymity                  1259 non-null   object
 17  leave                      1259 non-null   object
 18  mental_health_consequence  1259 non-null   object
 19  phys_health_consequence    1259 non-null   object
 20  coworkers                  1259 non-null   object
 21  supervisor                 1259 non-null   object
 22  mental_health_interview    1259 non-null   object
 23  phys_health_interview      1259 non-null   object
 24  mental_vs_physical         1259 non-null   object
 25  obs_consequence            1259 non-null   object
 26  comments                   164 non-null    object
dtypes: int64(1), object(26)
memory usage: 265.7+ KB
```

[6]:
```
print(df['Country'].value_counts())
print("\n \n")
print(df['state'].unique())
```

```
Country
United States             751
United Kingdom            185
Canada                     72
Germany                    45
Ireland                    27
Netherlands                27
Australia                  21
France                     13
India                      10
New Zealand                 8
Poland                      7
Switzerland                 7
Sweden                      7
Italy                       7
South Africa                6
Belgium                     6
Brazil                      6
Israel                      5
Singapore                   4
Bulgaria                    4
Austria                     3
Finland                     3
Mexico                      3
Russia                      3
Denmark                     2
Greece                      2
Colombia                    2
Croatia                     2
Portugal                    2
```

```
Moldova                      1
Georgia                      1
Bahamas, The                 1
China                        1
Thailand                     1
Czech Republic               1
Norway                       1
Romania                      1
Nigeria                      1
Japan                        1
Hungary                      1
Bosnia and Herzegovina       1
Uruguay                      1
Spain                        1
Zimbabwe                     1
Latvia                       1
Costa Rica                   1
Slovenia                     1
Philippines                  1
Name: count, dtype: int64
```

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

[7]:
```
df.drop(columns=['Timestamp', 'Country', 'state', 'comments'], inplace = True)
```

[8]:
```
print("The dataset contains different age groups including: \n")
print(df['Age'].unique())
print("\n \n")
print("The different gender notations used in our dataset are: \n")
print(df['Gender'].unique())
```

```
The dataset contains different age groups including:

[         37          44          32          31          33          35
          39          42          23          29          36          27
          46          41          34          30          40          38
          50          24          18          28          26          22
          19          25          45          21         -29          43
          56          60          54         329          55 99999999999
          48          20          57          58          47          62
          51          65          49       -1726           5          53
          61           8          11          -1          72]
```

```
The different gender notations used in our dataset are:

['Female' 'M' 'Male' 'male' 'female' 'm' 'Male-ish' 'maile' 'Trans-female'
 'Cis Female' 'F' 'something kinda male?' 'Cis Male' 'Woman' 'f' 'Mal'
 'Male (CIS)' 'queer/she/they' 'non-binary' 'Femake' 'woman' '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' 'cis male' 'A little about you' 'Malr' 'p' 'femail'
 'Cis Man' 'ostensibly male, unsure what that really means']
```

DATA PRE-PROCESSING AND VISUALIZATION

[9]:
```
df.drop(df[df['Age'] < 0].index, inplace = True)
df.drop(df[df['Age'] > 100].index, inplace = True)
df['Age'].unique()
```

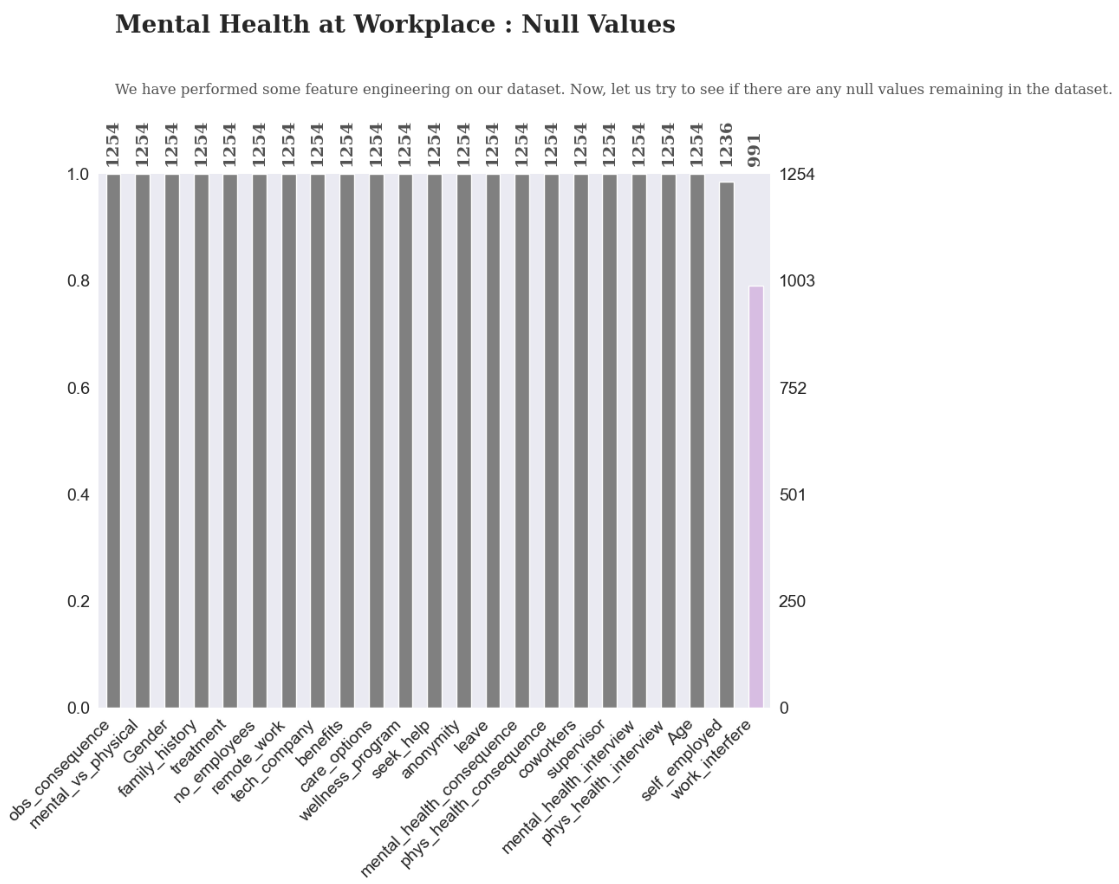
[9]:
```
array([37, 44, 32, 31, 33, 35, 39, 42, 23, 29, 36, 27, 46, 41, 34, 30, 40,
       38, 50, 24, 18, 28, 26, 22, 19, 25, 45, 21, 43, 56, 60, 54, 55, 48,
       20, 57, 58, 47, 62, 51, 65, 49,  5, 53, 61,  8, 11, 72],
      dtype=int64)
```

[10]:
```
df['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)

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

df["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',], 'Other', inplace =␣
 ↪True)

df['Gender'].value_counts()
```

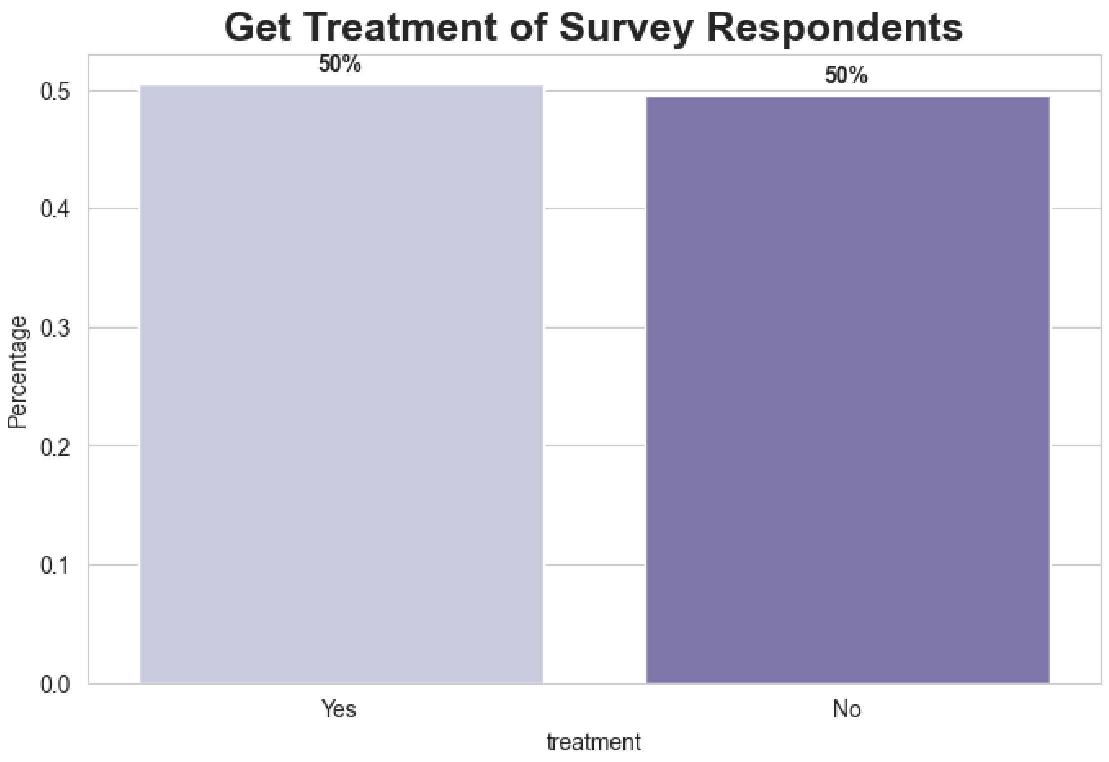
[10]:
```
Gender
Male      988
```

```
Female    247
Other      19
Name: count, dtype: int64
```

[11]:
```
sns.set_style('dark')
color =␣
  ↪['grey','grey','grey','grey','grey','grey','grey','grey','grey','grey','grey','grey',␣
  ↪'grey','grey','grey','grey','grey','grey',␣
  ↪'grey','grey','grey','grey','#D7BDE2']
msno.bar(df,fontsize =14, color = color, sort = 'descending', figsize = (10,8))

plt.text(0.05,1.265,'Mental Health at Workplace : Null Values', {'font':
  ↪'serif', 'size':20, 'weight':'bold'})
plt.text(0.05,1.15,'''We have performed some feature engineering on our dataset.
  ↪ Now, let us try to see if there are any null values remaining in the␣
  ↪dataset.''', {'font':'serif', 'size':12, 'weight':'normal'}, alpha = 0.8)
plt.xticks( rotation = 90,
                   **{'font':'serif','size':14,'weight':
  ↪'bold','horizontalalignment': 'center'},alpha = 0.8)
plt.show()
```

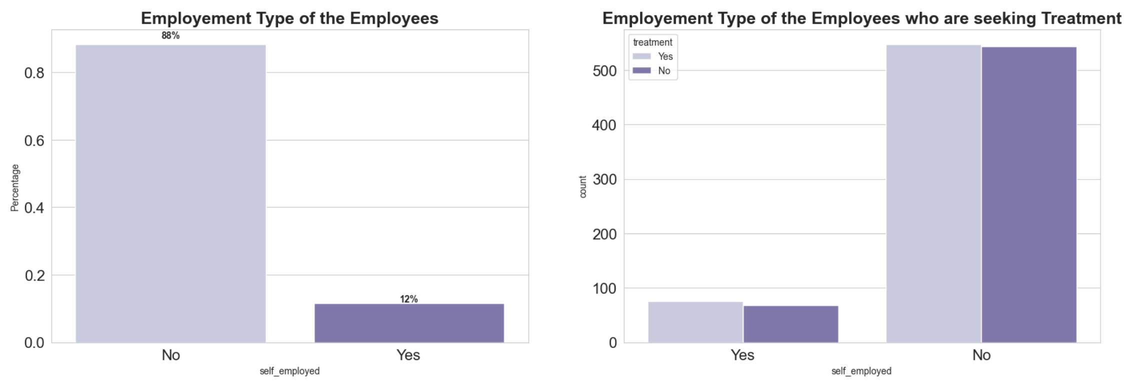


```
[12]: sns.set_style("whitegrid")
plt.figure(figsize = (8,5))
plt.title('Get Treatment of Survey Respondents', fontsize=18, fontweight='bold')
eda_percentage = df['treatment'].value_counts(normalize = True).
 ↪rename_axis('treatment').reset_index(name = 'Percentage')

ax = sns.barplot(x = 'treatment', y = 'Percentage', data = eda_percentage.
 ↪head(10), palette='Purples')
for p in ax.patches:
    width = p.get_width()
    height = p.get_height()
    x, y = p.get_xy()
    ax.annotate(f'{height:.0%}', (x + width/2, y + height*1.02), ha='center',␣
 ↪fontweight='bold')
```

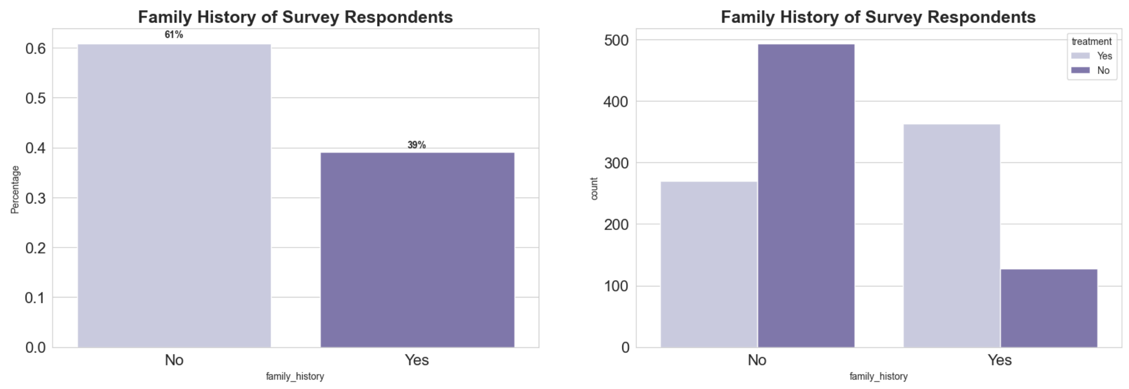


```
[13]: plt.figure(figsize = (20,6))
plt.subplot(1,2,1)
eda_percentage = df['self_employed'].value_counts(normalize = True).
 ↪rename_axis('self_employed').reset_index(name = 'Percentage')
ax = sns.barplot(x = 'self_employed', y = 'Percentage', data = eda_percentage,␣
 ↪palette = 'Purples')
```

```
for p in ax.patches:
    width = p.get_width()
    height = p.get_height()
    x, y = p.get_xy()
    ax.annotate(f'{height:.0%}', (x + width/2, y + height*1.02), ha='center',␣
 ↪fontweight='bold')
plt.title('Employement Type of the Employees', fontsize=18, fontweight='bold')
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.subplot(1,2,2)
sns.countplot(df['self_employed'], hue = df['treatment'], palette = 'Purples')
plt.title('Employement Type of the Employees who are seeking Treatment', ␣
 ↪fontsize=18, fontweight='bold')
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)

plt.show()
```

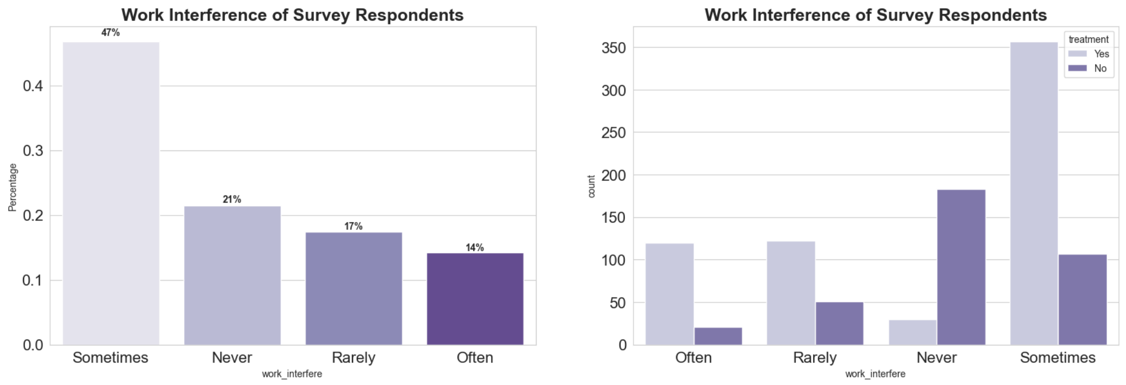


```
[14]: plt.figure(figsize = (20,6))
plt.subplot(1,2,1)
eda_percentage = df['family_history'].value_counts(normalize = True).
 ↪rename_axis('family_history').reset_index(name = 'Percentage')
ax = sns.barplot(x = 'family_history', y = 'Percentage', data = eda_percentage,␣
 ↪palette='Purples')
for p in ax.patches:
    width = p.get_width()
    height = p.get_height()
    x, y = p.get_xy()
    ax.annotate(f'{height:.0%}', (x + width/2, y + height*1.02), ha='center',␣
 ↪fontweight='bold')
plt.title('Family History of Survey Respondents', fontsize=18,␣
 ↪fontweight='bold')
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.subplot(1,2,2)
sns.countplot(df['family_history'], hue = df['treatment'], palette='Purples')
plt.title('Family History of Survey Respondents', fontsize=18,␣
  ↪fontweight='bold')
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)

plt.show()
```



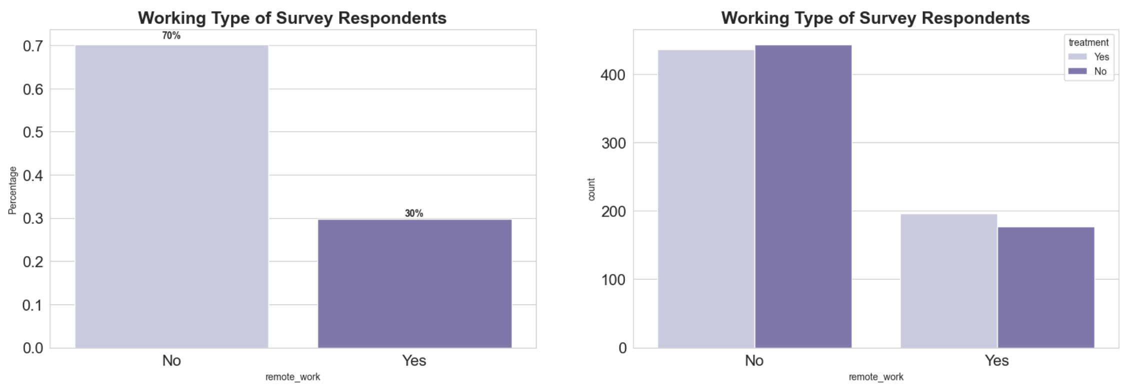
```
[15]: plt.figure(figsize = (20,6))
plt.subplot(1,2,1)
eda_percentage = df['work_interfere'].value_counts(normalize = True).
  ↪rename_axis('work_interfere').reset_index(name = 'Percentage')
ax = sns.barplot(x = 'work_interfere', y = 'Percentage', data = eda_percentage,␣
  ↪palette='Purples')
for p in ax.patches:
    width = p.get_width()
    height = p.get_height()
    x, y = p.get_xy()
    ax.annotate(f'{height:.0%}', (x + width/2, y + height*1.02), ha='center',␣
  ↪fontweight='bold')
plt.title('Work Interference of Survey Respondents', fontsize=18,␣
  ↪fontweight='bold')
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.subplot(1,2,2)
sns.countplot(df['work_interfere'], hue = df['treatment'], palette = 'Purples')
plt.title('Work Interference of Survey Respondents', fontsize=18,␣
  ↪fontweight='bold')
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
```

```
[15]: (array([  0.,  50., 100., 150., 200., 250., 300., 350., 400.]),
       [Text(0, 0.0, '0'),
        Text(0, 50.0, '50'),
        Text(0, 100.0, '100'),
        Text(0, 150.0, '150'),
        Text(0, 200.0, '200'),
        Text(0, 250.0, '250'),
        Text(0, 300.0, '300'),
        Text(0, 350.0, '350'),
        Text(0, 400.0, '400')])
```



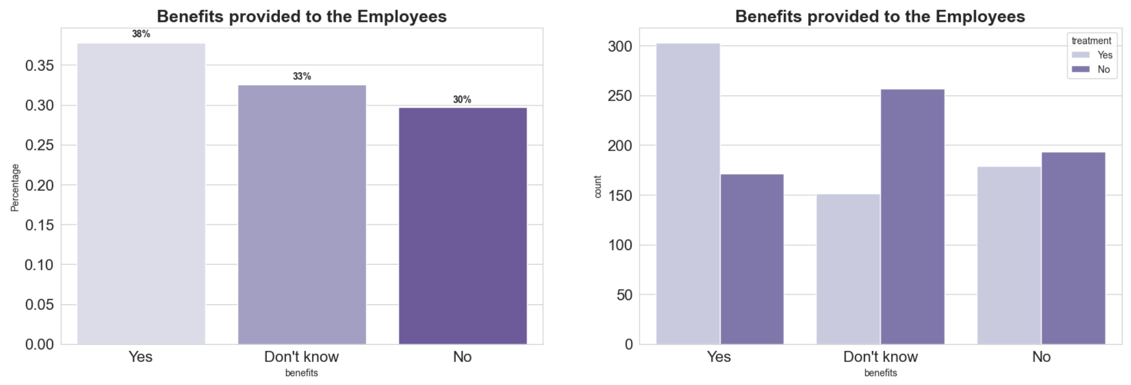
```
[16]: plt.figure(figsize = (20,6))
      plt.subplot(1,2,1)
      eda_percentage = df['remote_work'].value_counts(normalize = True).
       ↪rename_axis('remote_work').reset_index(name = 'Percentage')
      ax = sns.barplot(x = 'remote_work', y = 'Percentage', data = eda_percentage,␣
       ↪palette='Purples')
      for p in ax.patches:
          width = p.get_width()
          height = p.get_height()
          x, y = p.get_xy()
          ax.annotate(f'{height:.0%}', (x + width/2, y + height*1.02), ha='center',␣
       ↪fontweight='bold')

      plt.title('Working Type of Survey Respondents', fontsize=18, fontweight='bold')
      plt.xticks(fontsize=16)
      plt.yticks(fontsize=16)
      plt.subplot(1,2,2)
      sns.countplot(df['remote_work'], hue = df['treatment'], palette='Purples')
      plt.title('Working Type of Survey Respondents', fontsize=18, fontweight='bold')
      plt.xticks(fontsize=16)
      plt.yticks(fontsize=16)
```

```
[16]: (array([  0., 100., 200., 300., 400., 500.]),
        [Text(0, 0.0, '0'),
         Text(0, 100.0, '100'),
         Text(0, 200.0, '200'),
         Text(0, 300.0, '300'),
         Text(0, 400.0, '400'),
         Text(0, 500.0, '500')])
```



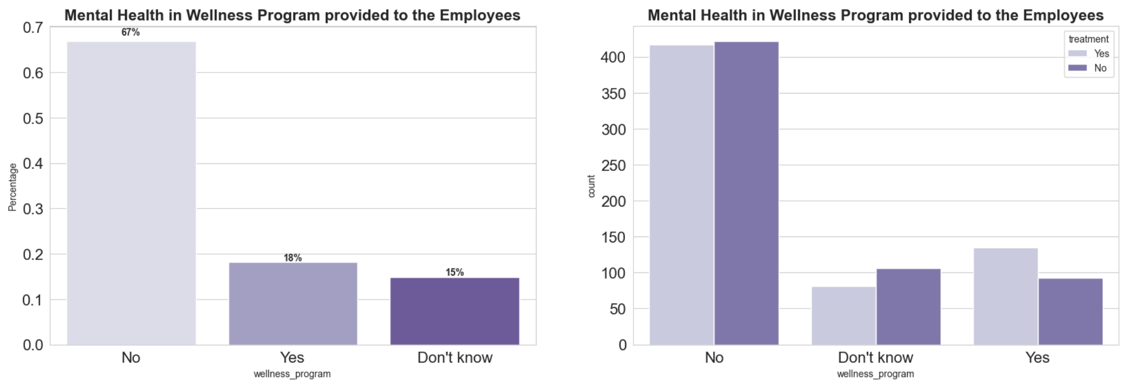
```
[17]: plt.figure(figsize = (20,6))
      plt.subplot(1,2,1)
      eda_percentage = df['benefits'].value_counts(normalize = True).
       ↪rename_axis('benefits').reset_index(name = 'Percentage')
      ax = sns.barplot(x = 'benefits', y = 'Percentage', data = eda_percentage,␣
       ↪palette='Purples')
      for p in ax.patches:
          width = p.get_width()
          height = p.get_height()
          x, y = p.get_xy()
          ax.annotate(f'{height:.0%}', (x + width/2, y + height*1.02), ha='center',␣
       ↪fontweight='bold')
      plt.title('Benefits provided to the Employees', fontsize=18, fontweight='bold')
      plt.xticks(fontsize=16)
      plt.yticks(fontsize=16)
      plt.subplot(1,2,2)
      sns.countplot(df['benefits'], hue = df['treatment'], palette='Purples')
      plt.title('Benefits provided to the Employees', fontsize=18, fontweight='bold')
      plt.xticks(fontsize=16)
      plt.yticks(fontsize=16)
```

```
[17]: (array([  0.,  50., 100., 150., 200., 250., 300., 350.]),
        [Text(0, 0.0, '0'),
         Text(0, 50.0, '50'),
         Text(0, 100.0, '100'),
         Text(0, 150.0, '150'),
         Text(0, 200.0, '200'),
         Text(0, 250.0, '250'),
         Text(0, 300.0, '300'),
         Text(0, 350.0, '350')])
```



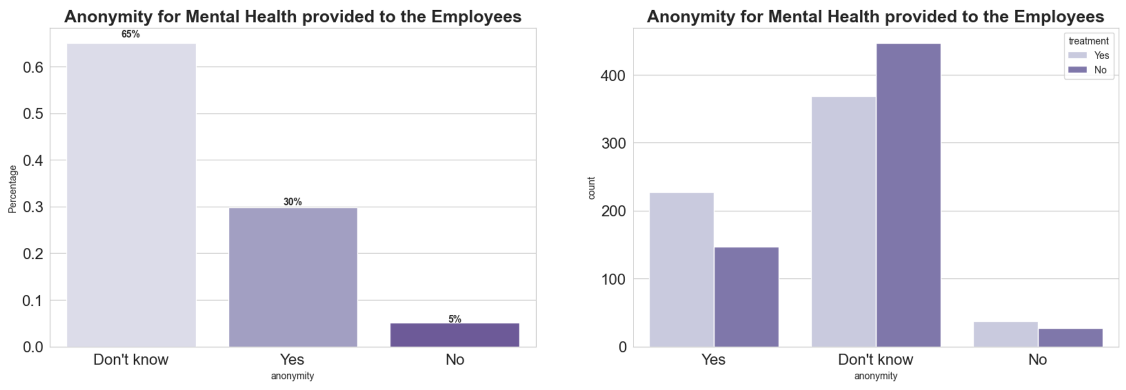
```
[19]: plt.figure(figsize=(20,6))
      plt.subplot(1,2,1)
      eda_percentage = df['wellness_program'].value_counts(normalize = True).
       ↪rename_axis('wellness_program').reset_index(name = 'Percentage')
      ax = sns.barplot(x = 'wellness_program', y = 'Percentage', data =␣
       ↪eda_percentage, palette='Purples')
      for p in ax.patches:
          width = p.get_width()
          height = p.get_height()
          x, y = p.get_xy()
          ax.annotate(f'{height:.0%}', (x + width/2, y + height*1.02), ha='center',␣
       ↪fontweight='bold')
      plt.title('Mental Health in Wellness Program provided to the Employees',␣
       ↪fontsize=16, fontweight='bold')
      plt.xticks(fontsize=16)
      plt.yticks(fontsize=16)
      plt.subplot(1,2,2)
      sns.countplot(df['wellness_program'], hue = df['treatment'], palette='Purples')
      plt.title('Mental Health in Wellness Program provided to the Employees',␣
       ↪fontsize=16, fontweight='bold')
      plt.xticks(fontsize=16)
      plt.yticks(fontsize=16)
```

```
[19]: (array([  0.,  50., 100., 150., 200., 250., 300., 350., 400., 450.]),
       [Text(0, 0.0, '0'),
        Text(0, 50.0, '50'),
        Text(0, 100.0, '100'),
        Text(0, 150.0, '150'),
        Text(0, 200.0, '200'),
        Text(0, 250.0, '250'),
        Text(0, 300.0, '300'),
        Text(0, 350.0, '350'),
        Text(0, 400.0, '400'),
        Text(0, 450.0, '450')])
```



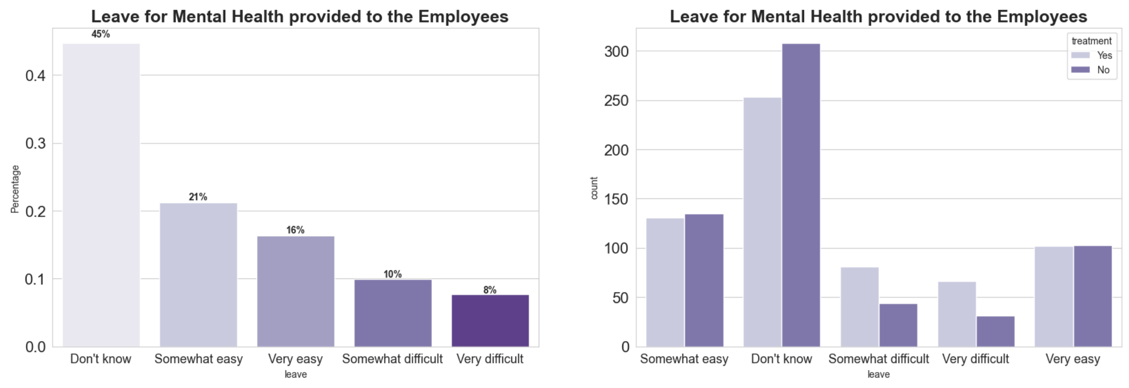
[20]:
```
plt.figure(figsize=(20,6))
plt.subplot(1,2,1)
eda_percentage = df['anonymity'].value_counts(normalize = True).
 ↪rename_axis('anonymity').reset_index(name = 'Percentage')
ax = sns.barplot(x = 'anonymity', y = 'Percentage', data = eda_percentage,␣
 ↪palette='Purples')
for p in ax.patches:
    width = p.get_width()
    height = p.get_height()
    x, y = p.get_xy()
    ax.annotate(f'{height:.0%}', (x + width/2, y + height*1.02), ha='center',␣
 ↪fontweight='bold')
plt.title('Anonymity for Mental Health provided to the Employees', fontsize=18,␣
 ↪fontweight='bold')
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.subplot(1,2,2)
sns.countplot(df['anonymity'], hue = df['treatment'], palette='Purples')
plt.title('Anonymity for Mental Health provided to the Employees', fontsize=18,␣
 ↪fontweight='bold')
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
```

[20]:
```
(array([  0., 100., 200., 300., 400., 500.]),
 [Text(0, 0.0, '0'),
  Text(0, 100.0, '100'),
  Text(0, 200.0, '200'),
  Text(0, 300.0, '300'),
  Text(0, 400.0, '400'),
  Text(0, 500.0, '500')])
```



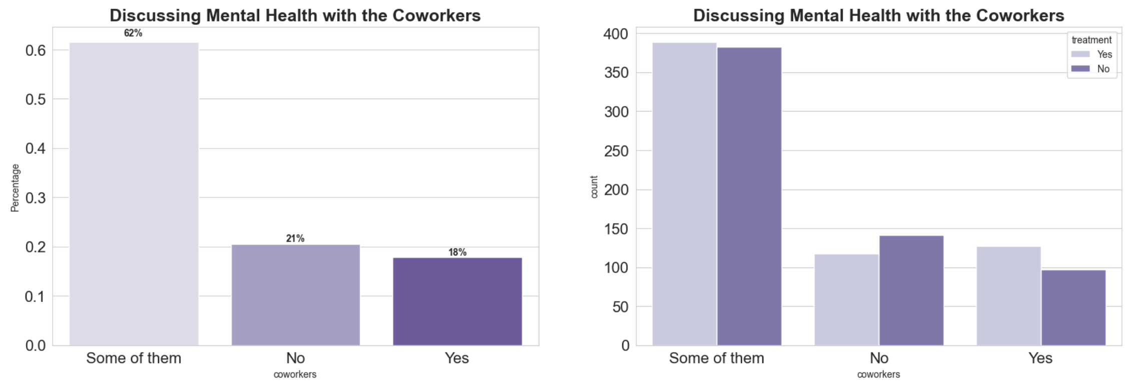
[21]:
```
plt.figure(figsize=(20,6))
plt.subplot(1,2,1)
eda_percentage = df['leave'].value_counts(normalize = True).
 ↪rename_axis('leave').reset_index(name = 'Percentage')
ax = sns.barplot(x = 'leave', y = 'Percentage', data = eda_percentage,␣
 ↪palette='Purples')
for p in ax.patches:
    width = p.get_width()
    height = p.get_height()
    x, y = p.get_xy()
    ax.annotate(f'{height:.0%}', (x + width/2, y + height*1.02), ha='center',␣
 ↪fontweight='bold')
plt.title('  Leave for Mental Health provided to the Employees', fontsize=18,␣
 ↪fontweight='bold')
plt.xticks(fontsize=13)
plt.yticks(fontsize=16)
plt.subplot(1,2,2)
sns.countplot(df['leave'], hue = df['treatment'], palette='Purples')
plt.title('Leave for Mental Health provided to the Employees', fontsize=18,␣
 ↪fontweight='bold')
plt.xticks(fontsize=13)
plt.yticks(fontsize=16)
```

[21]:
```
(array([  0.,  50., 100., 150., 200., 250., 300., 350.]),
 [Text(0, 0.0, '0'),
  Text(0, 50.0, '50'),
  Text(0, 100.0, '100'),
  Text(0, 150.0, '150'),
  Text(0, 200.0, '200'),
  Text(0, 250.0, '250'),
  Text(0, 300.0, '300'),
  Text(0, 350.0, '350')])
```



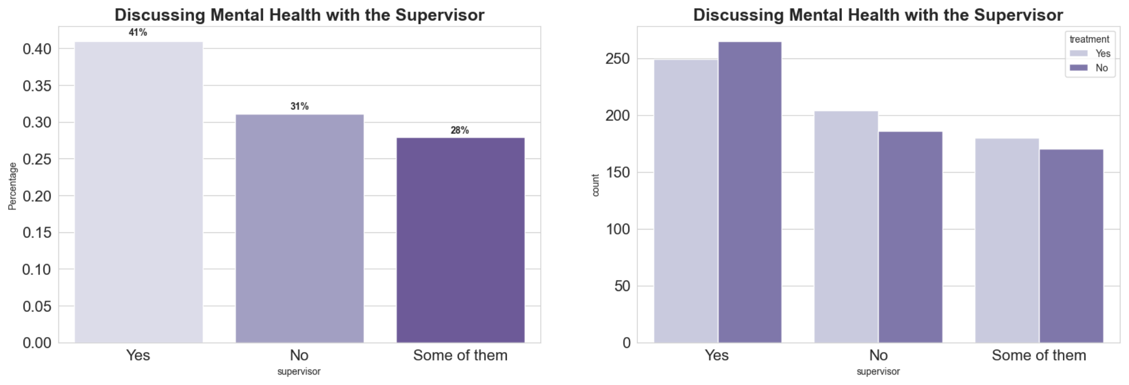
[22]:
```
plt.figure(figsize=(20,6))
plt.subplot(1,2,1)
eda_percentage = df['coworkers'].value_counts(normalize = True).
 ↪rename_axis('coworkers').reset_index(name = 'Percentage')
ax = sns.barplot(x = 'coworkers', y = 'Percentage', data = eda_percentage,␣
 ↪palette='Purples')
for p in ax.patches:
    width = p.get_width()
    height = p.get_height()
    x, y = p.get_xy()
    ax.annotate(f'{height:.0%}', (x + width/2, y + height*1.02), ha='center',␣
 ↪fontweight='bold')

plt.title('Discussing Mental Health with the Coworkers', fontsize=18,␣
 ↪fontweight='bold')
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.subplot(1,2,2)
sns.countplot(df['coworkers'], hue = df['treatment'], palette='Purples')
plt.title('Discussing Mental Health with the Coworkers', fontsize=18,␣
 ↪fontweight='bold')
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
```

[22]:
```
(array([  0.,  50., 100., 150., 200., 250., 300., 350., 400., 450.]),
 [Text(0, 0.0, '0'),
  Text(0, 50.0, '50'),
  Text(0, 100.0, '100'),
  Text(0, 150.0, '150'),
  Text(0, 200.0, '200'),
  Text(0, 250.0, '250'),
  Text(0, 300.0, '300'),
  Text(0, 350.0, '350'),
  Text(0, 400.0, '400'),
  Text(0, 450.0, '450')])
```



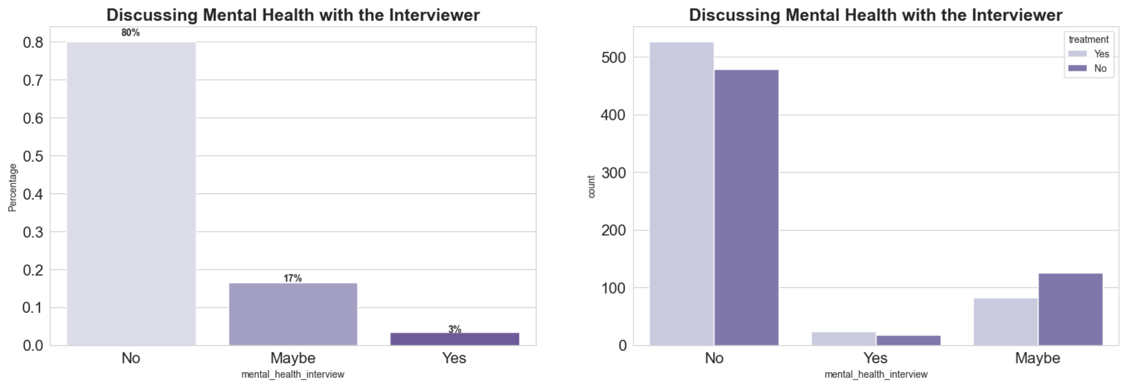
[23]:
```
plt.figure(figsize=(20,6))
plt.subplot(1,2,1)
eda_percentage = df['supervisor'].value_counts(normalize = True).
 ↪rename_axis('supervisor').reset_index(name = 'Percentage')
ax = sns.barplot(x = 'supervisor', y = 'Percentage', data = eda_percentage,␣
 ↪palette='Purples')
for p in ax.patches:
    width = p.get_width()
    height = p.get_height()
    x, y = p.get_xy()
    ax.annotate(f'{height:.0%}', (x + width/2, y + height*1.02), ha='center',␣
 ↪fontweight='bold')

plt.title('Discussing Mental Health with the Supervisor', fontsize=18,␣
 ↪fontweight='bold')
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.subplot(1,2,2)
sns.countplot(df['supervisor'], hue = df['treatment'], palette='Purples')
plt.title('Discussing Mental Health with the Supervisor', fontsize=18,␣
 ↪fontweight='bold')
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
```

```
[23]: (array([  0.,  50., 100., 150., 200., 250., 300.]),
       [Text(0, 0.0, '0'),
        Text(0, 50.0, '50'),
        Text(0, 100.0, '100'),
        Text(0, 150.0, '150'),
        Text(0, 200.0, '200'),
        Text(0, 250.0, '250'),
        Text(0, 300.0, '300')])
```



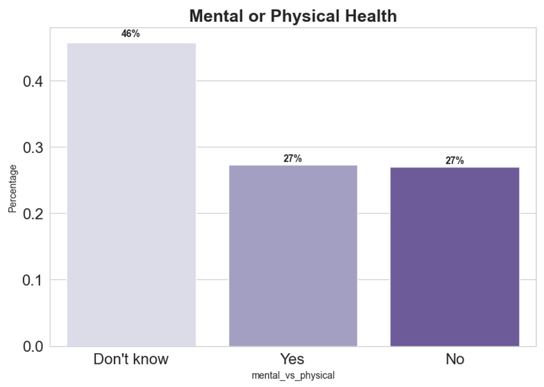
```
[24]: plt.figure(figsize=(20,6))
      plt.subplot(1,2,1)
      eda_percentage = df['mental_health_interview'].value_counts(normalize = True).
       ↪rename_axis('mental_health_interview').reset_index(name = 'Percentage')
      ax = sns.barplot(x = 'mental_health_interview', y = 'Percentage', data =␣
       ↪eda_percentage, palette='Purples')
      for p in ax.patches:
          width = p.get_width()
          height = p.get_height()
          x, y = p.get_xy()
          ax.annotate(f'{height:.0%}', (x + width/2, y + height*1.02), ha='center',␣
       ↪fontweight='bold')
      plt.title('Discussing Mental Health with the Interviewer', fontsize=18,␣
       ↪fontweight='bold')
      plt.xticks(fontsize=16)
      plt.yticks(fontsize=16)
      plt.subplot(1,2,2)
      sns.countplot(df['mental_health_interview'], hue = df['treatment'],␣
       ↪palette='Purples')
      plt.title('Discussing Mental Health with the Interviewer', fontsize=18,␣
       ↪fontweight='bold')
      plt.xticks(fontsize=16)
      plt.yticks(fontsize=16)
```

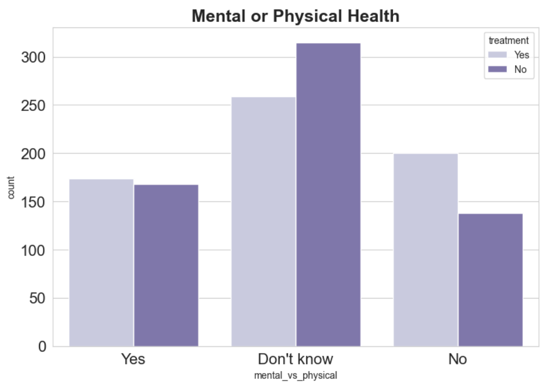
```
[24]: (array([  0., 100., 200., 300., 400., 500., 600.]),
       [Text(0, 0.0, '0'),
        Text(0, 100.0, '100'),
        Text(0, 200.0, '200'),
        Text(0, 300.0, '300'),
        Text(0, 400.0, '400'),
        Text(0, 500.0, '500'),
        Text(0, 600.0, '600')])
```



```
[25]: plt.figure(figsize=(20,6))
      plt.subplot(1,2,1)
      eda_percentage = df['mental_vs_physical'].value_counts(normalize = True).
       ↪rename_axis('mental_vs_physical').reset_index(name = 'Percentage')
      ax = sns.barplot(x = 'mental_vs_physical', y = 'Percentage', data =␣
       ↪eda_percentage, palette='Purples')
      for p in ax.patches:
          width = p.get_width()
          height = p.get_height()
          x, y = p.get_xy()
          ax.annotate(f'{height:.0%}', (x + width/2, y + height*1.02), ha='center',␣
       ↪fontweight='bold')
      plt.title('Mental or Physical Health', fontsize=18, fontweight='bold')
      plt.xticks(fontsize=16)
      plt.yticks(fontsize=16)
      plt.subplot(1,2,2)
      sns.countplot(df['mental_vs_physical'], hue = df['treatment'],␣
       ↪palette='Purples')
      plt.title('Mental or Physical Health', fontsize=18, fontweight='bold')
      plt.xticks(fontsize=16)
      plt.yticks(fontsize=16)
```

```
[25]: (array([  0.,  50., 100., 150., 200., 250., 300., 350.]),
       [Text(0, 0.0, '0'),
        Text(0, 50.0, '50'),
        Text(0, 100.0, '100'),
        Text(0, 150.0, '150'),
        Text(0, 200.0, '200'),
        Text(0, 250.0, '250'),
        Text(0, 300.0, '300'),
        Text(0, 350.0, '350')])
```





[ ]: