DSCI 6007-01 Final Project

on

"Mental Health at Workplace"

Master of Science in Data Science (DS)

TEAM MEMBERS:	contact:
Gowthami Middela	gmidd1@unh.newhaven.edu
Ashish Manchala	amanc9@unh.newhaven.edu
Yasaswi Nallamothu	ynall1@unh.newhaven.edu
Gangai kaliprasad Battula	gbatt1@unh.newhaven.edu

Contents

Executive Summary

Mental health is a critical issue that is only now gaining traction in our community. Data science is becoming increasingly common in the healthcare business as new data sources and analysis tools are developed. There is a significant unmet need in the globe for mental health services. Data science can help us better understand mental health challenges. We may be able to employ data analytics to better understand and implement therapies for mental health disorders.

Abstract

In this study, we will attempt to comprehend the elements that contribute to a person's mental health. This project makes use of a dataset from Kaggle that was taken from a 2014 survey that analyzes attitudes toward mental health and the prevalence of mental health issues in the workplace. This study allows us to gain a systematic understanding of mental health in the workplace. All files are stored in this Github Repository

Topics in CRISP-DM Method (Cross Industry Standard Process)

Iteration 1:

The process involved during the first iteration are:

1. Business Understanding
2. Data Understanding
3. Data Preparation
4. Modeling
5. Evaluation and
6. Deployment

Iteration 2:

Goal Redefinition

- Data Understanding
- Modeling and Evaluation

Iteration 3:

Variable and Instance Selection

Review of available research

Since the beginning of the covid pandemic, most work institutions around the world have adopted an online model, which creates additional work pressure and mental stress due to work and family management.

Many healthcare workers maintain stigmatizing views within mental healthcare settings, which has been highlighted as a key obstacle to access treatment and recovery, as well as inferior quality physical care for those with mental illnesses. Fellows should be encouraged to communicate their psychological discomfort in secure areas, and leaders should acknowledge their worries. The medical community's chronic stigma around mental illness, including interpersonal and self-stigma, sometimes discourages professionals from seeking assistance.

The absence of information on the operation and efficacy of youth-focused mental health services makes it difficult to analyze and, hence, improve them. Manufacturing enterprises in a healthy society are accountable for more than just generating lucrative amounts of goods and services, and their managers understand that effective management leads to higher output. This critical aim, however, cannot be realized without a commitment to and conviction in the workforce's mental health. As a result, one of the tasks of any effective, astute, and resourceful manager is to monitor the mental health of team members.

Bipolar illness is one of the most common types of depression among all mental health issues. A survey was performed among individuals to collect data on mental health. These findings were utilized by the machine learning algorithm, which was believed to improve the validity of the ML technique. With advancements in the medical field and the use of social networks in the field to detect in the field. To be successful, the clinical factors must be combined utilizing database management technologies such as big data. MongoDB is one of the technologies used to manage massive data, extract information, and provide accurate results in treating a variety of mental illness problems at a cheap cost and with great efficiency.

Challenge

How to handle mental health at the workplace

Mental Health affects mental, psychological, and social well-being. It also affects how we think, feel and act. It also helps us in determining how we handle stress by relating it to others. The impact of mental health on an organization can mean a lot of things: an increase in absent days from work, Decrease in productivity. In the US, approximately 70 percent of adults with depression are in the workforce. Employees with depression will miss a lot of days.

Solution

This problem can be solved using Mental Health First Aid. It helps participants to notice and support individuals who are suffering from mental health. It teaches employees communication and support skills which can help people suffering from mental health.

Research shows that employees who used first aid have increased awareness of mental health among themselves and their co-workers. It allows them to recognize the signs of someone who is struggling with mental health and teaches them the skills to when and where to reach out.

Moreover, they conduct an Employee Assistance Program that focuses on mental and physical health. These measures can help create a healthy and productive work environment that reduces the stigma associated with mental illness.

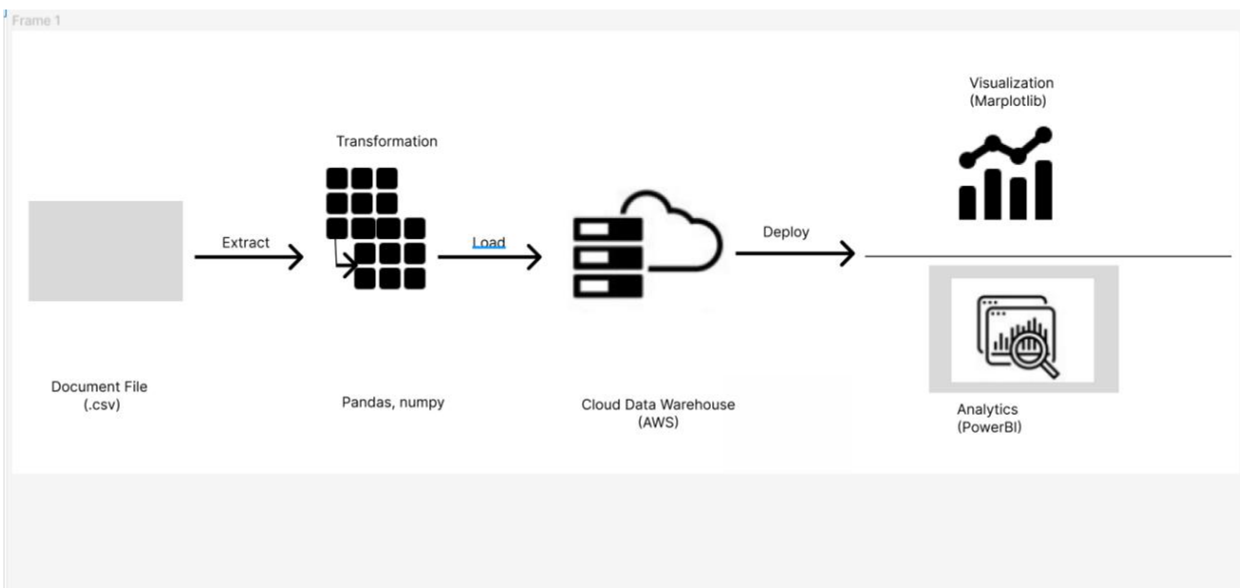
```
df['Country'].value_counts()
```

```
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
```

```
df['state'].unique()
```

```
array(['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'], dtype=object)
```

Graphical View of ETL Tools :



Data

- There are a total of 26 columns in the dataset.
- We see that except the age column, all the columns are of object datatype.

- Comment column seems to contain most number (70%) of null values, which makes sense because it was an optional text box so it's reasonable to expect that many (most) respondents would leave it blank.
- We will be dropping the timestamp column because it's contains date, month, year and time the respondent took this questionnaire, which is irrelevant for us.

The state column also contains a lot of null values. We'll dig deeper into that

	Timestamp	Age	Gender	Country	state	self_employed	family_history	treatment	work_interfere	no_employees	...	leave	mental_health_consequence
0	2014-08-27 11:29:31	37	Female	United States	IL	NaN	No	Yes	Often	6-25	...	Somewhat easy	No
1	2014-08-27 11:29:37	44	M	United States	IN	NaN	No	No	Rarely	More than 1000	...	Don't know	Maybe
2	2014-08-27 11:29:44	32	Male	Canada	NaN	NaN	No	No	Rarely	6-25	...	Somewhat difficult	No
3	2014-08-27 11:29:46	31	Male	United Kingdom	NaN	NaN	Yes	Yes	Often	26-100	...	Somewhat difficult	Yes
4	2014-08-27 11:30:22	31	Male	United States	TX	NaN	No	No	Never	100-500	...	Don't know	No

phys_health_consequence	coworkers	supervisor	mental_health_interview	phys_health_interview	mental_vs_physical	obs_consequence	comments
No	Some of them	Yes	No	Maybe	Yes	No	NaN
No	No	No	No	No	Don't know	No	NaN
No	Yes	Yes	Yes	Yes	No	No	NaN
Yes	Some of them	No	Maybe	Maybe	No	Yes	NaN
No	Some of them	Yes	Yes	Yes	Don't know	No	NaN

In [4]:
```
# Describing the dataset.
df.head()
print("   ")
df.dtypes
print("   ")
df.isnull().sum()
```

Out[4]:

	Timestamp	Age	Gender	Country	state	self_employed	family_history	treatment	work_interfere	no_employees	...	leave	mental_health_consequence
0	2014-08-27 11:29:31	37	Female	United States	IL	NaN	No	Yes	Often	6-25	...	Somewhat easy	No
1	2014-08-27 11:29:37	44	M	United States	IN	NaN	No	No	Rarely	More than 1000	...	Don't know	Maybe
2	2014-08-27 11:29:44	32	Male	Canada	NaN	NaN	No	No	Rarely	6-25	...	Somewhat difficult	No
3	2014-08-27 11:29:46	31	Male	United Kingdom	NaN	NaN	Yes	Yes	Often	26-100	...	Somewhat difficult	Yes
4	2014-08-27 11:30:22	31	Male	United States	TX	NaN	No	No	Never	100-500	...	Don't know	No

```
Out[4]: Timestamp                    object
        Age                           int64
        Gender                       object
        Country                      object
        state                        object
        self_employed                object
        family_history               object
        treatment                    object
        work_interfere               object
        no_employees                 object
        remote_work                  object
        tech_company                 object
        benefits                     object
        care_options                 object
        wellness_program             object
        seek_help                    object
        anonymity                    object
        leave                        object
        mental_health_consequence    object
        phys_health_consequence      object
        coworkers                    object
        supervisor                   object
        mental_health_interview      object
        phys_health_interview        object
        mental_vs_physical           object
        obs_consequence              object
        comments                     object
        dtype: object
```

```
Out[4]: 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
```

Comments column have a lot of null values as it's an optional text box and many of them may leave it blank. Timestamp column is also not required because it contains time when the respondent took the questionnaire. State column has lot of null values.¶

It will be really misleading to conclude that one country faces more problems with mental health of employees as 60% of people are from the US.

```
In [6]: # Let's drop Timestamp,Country,state,comments columns.
        df.drop(columns=['Timestamp', 'Country', 'state', 'comments'],inplace=True)
```

Data Preparation and Feature Engineering :

```
In [7]: print("Different age groups used in the dataset : ")
        df['Age'].unique()
        print("Different gender notations used in the dataset :")
        df['Gender'].unique()

        Different age groups used in the dataset :

Out[7]: array([          37,          44,          32,          31,          33,
                         35,          39,          42,          23,          29,
                         36,          27,          46,          41,          34,
                         30,          40,          38,          50,          24,
                         18,          28,          26,          22,          19,
                         25,          45,          21,         -29,          43,
                         56,          60,          54,         329,          55,
                99999999999,          48,          20,          57,          58,
                         47,          62,          51,          65,          49,
                      -1726,           5,          53,          61,           8,
                         11,          -1,          72])

        Different gender notations used in the dataset :

Out[7]: array(['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'], dtype=object)
```

How can age be negative? And how can the age be less than 20? Are they allowed to even work?

Regarding the gender column Males and Females have described themselves in so many different ways. This is what happens when we don't have check boxes or radio buttons while taking surveys.

```
In [8]: df.drop(df[df['Age'] < 20].index, inplace = True)
        df.drop(df[df['Age'] > 100].index, inplace = True)
        df['Age'].unique()

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

```
In [9]: 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()

Out[9]: Male      973
        Female    246
        Other      16
        Name: Gender, dtype: int64
```

From this output we can conclude that the number of males in the tech industry are more when compared to females.

```
In [29]: # Checking if there are any null-values.
         df.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 1235 entries, 0 to 1258
Data columns (total 23 columns):
 #   Column                     Non-Null Count  Dtype 
---  ------                     --------------  ----- 
 0   Age                        1235 non-null   int64 
 1   Gender                     1235 non-null   object
 2   self_employed              1217 non-null   object
 3   family_history             1235 non-null   object
 4   treatment                  1235 non-null   object
 5   work_interfere             978 non-null    object
 6   no_employees               1235 non-null   object
 7   remote_work                1235 non-null   object
 8   tech_company               1235 non-null   object
 9   benefits                   1235 non-null   object
 10  care_options               1235 non-null   object
 11  wellness_program           1235 non-null   object
 12  seek_help                  1235 non-null   object
 13  anonymity                  1235 non-null   object
 14  leave                      1235 non-null   object
 15  mental_health_consequence  1235 non-null   object
 16  phys_health_consequence    1235 non-null   object
 17  coworkers                  1235 non-null   object
 18  supervisor                 1235 non-null   object
 19  mental_health_interview    1235 non-null   object
 20  phys_health_interview      1235 non-null   object
 21  mental_vs_physical         1235 non-null   object
 22  obs_consequence            1235 non-null   object
dtypes: int64(1), object(22)
memory usage: 263.9+ KB
```

```
In [30]: # Replacing the null-values.
         df['work_interfere'] = df['work_interfere'].fillna('Don\'t know' )
         print(df['work_interfere'].unique())

         ['Often' 'Rarely' 'Never' 'Sometimes' "Don't know"]
```

```
In [31]: df['self_employed'] = df['self_employed'].fillna('No')
         print(df['self_employed'].unique())

         ['No' 'Yes']
```

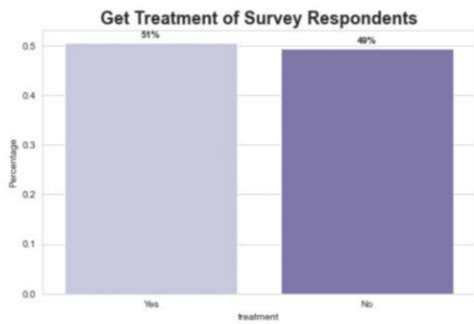
```
In [32]: df.isnull().sum()

Out[32]: Age                          0
         Gender                       0
         self_employed                0
         family_history               0
         treatment                    0
         work_interfere               0
         no_employees                 0
         remote_work                  0
         tech_company                 0
         benefits                     0
         care_options                 0
         wellness_program             0
         seek_help                    0
         anonymity                    0
         leave                        0
         mental_health_consequence    0
         phys_health_consequence      0
         coworkers                    0
         supervisor                   0
         mental_health_interview      0
         phys_health_interview        0
         mental_vs_physical           0
         obs_consequence              0
         dtype: int64
```

Since most of the columns except Age column are of object type we perform one-hot encoding.

```
In [34]: # Since most of the columns except Age column are of object type we perform one-hot encoding.
         from sklearn.preprocessing import LabelEncoder
         object_cols = ['Gender', 'self_employed', 'family_history', 'treatment',
                'work_interfere', 'no_employees', 'remote_work', 'tech_company',
                'benefits', 'care_options', 'wellness_program', 'seek_help',
                'anonymity', 'leave', 'mental_health_consequence',
                'phys_health_consequence', 'coworkers', 'supervisor',
                'mental_health_interview', 'phys_health_interview',
                'mental_vs_physical', 'obs_consequence']
         label_encoder = LabelEncoder()
         for col in object_cols:
             label_encoder.fit(df[col])
             df[col] = label_encoder.transform(df[col])
```

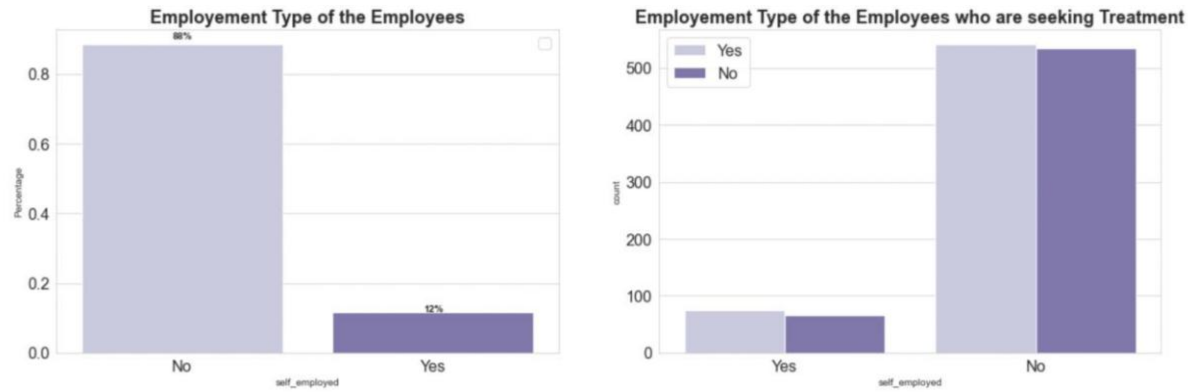
Exploratory Data Analysis :

```
In [10]: 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')

Out[10]: <Figure size 576x360 with 0 Axes>

Out[10]: Text(0.5, 1.0, 'Get Treatment of Survey Respondents')

Out[10]: Text(0.0, 0.5161943319838057, '51%')

Out[10]: Text(1.0, 0.5038056680161943, '49%')
```



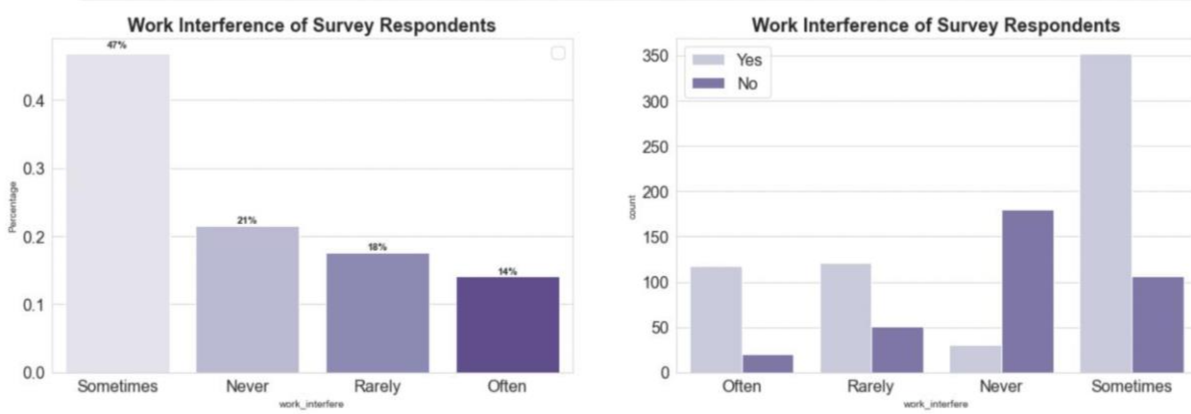
From this graph we can infer the percentage of respondents who want to get treatment is 50%. If employees have good mental health they can : Be more productive, Take active participation in employee engagement activities etc.

```
In [11]: 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 = 'Pe
         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.legend(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.legend(fontsize=16)

         plt.show()
```



Most of the working class employees responded to the survey when compared to working class. But we can observe from the second graph that both classes have the same percentage of people who are seeking treatment.

```
In [13]: 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 = '
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.legend(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)
plt.legend(fontsize=16)
```



Around 70 percent of employees don't work remotely hence most of the mental health issues occur at the workplace. Number of employees seeking treatment in both the categories are more or equal.

Creating and Evaluating the Models :

In [39]:
```
# Splitting the dataset into train and test datasets.
from sklearn.model_selection import train_test_split
X = df.drop('treatment', axis = 1)
y = df['treatment']

X_train, X_test, y_train, y_test = train_test_split(X,y,
                                                    stratify = y,
                                                     test_size = 0.3,
                                                    random_state = 101)
```

In [40]:
```
%%time
LR = LogisticRegression(solver='lbfgs', max_iter=10000, random_state=555)
RF = RandomForestClassifier(n_estimators = 100, random_state=555)
SVM = SVC(random_state=0, probability=True)
KNC = KNeighborsClassifier()
DTC = DecisionTreeClassifier()
ABC = AdaBoostClassifier(n_estimators = 100)
BC = BaggingClassifier(n_estimators = 100)
GBC = GradientBoostingClassifier(n_estimators = 100)
clf_XGB = XGBClassifier(n_estimators = 100, seed=555,eval_metric='logloss')
clfs = []
print('5-fold cross validation:\n')
for clf, label in zip([LR, RF, KNC, DTC, ABC, BC, GBC, clf_XGB],
                      ['Logistic Regression',
                       'Random Forest',
                       'KNeighbors',
                       'Decision Tree',
                       'Ada Boost',
                       'Bagging',
                       'Gradient Boosting',
                       'XGBoost']):
    scores = sklearn.model_selection.cross_val_score(clf, X_train, y_train, cv=5, scoring="accuracy")
    print("Train CV Accuracy: %0.3f (+/- %0.3f) [%s]" % (scores.mean(), scores.std(), label))
    md = clf.fit(X_train, y_train)
    clfs.append(md)
    print("Test Accuracy: %0.4f " % (sklearn.metrics.accuracy_score(clf.predict(X_test), y_test)))
```

```
5-fold cross validation:

Train CV Accuracy: 0.788 (+/- 0.062) [Logistic Regression]
Test Accuracy: 0.8194
Train CV Accuracy: 0.811 (+/- 0.044) [Random Forest]
Test Accuracy: 0.8275
Train CV Accuracy: 0.753 (+/- 0.034) [KNeighbors]
Test Accuracy: 0.7682
Train CV Accuracy: 0.749 (+/- 0.025) [Decision Tree]
Test Accuracy: 0.7493
Train CV Accuracy: 0.800 (+/- 0.039) [Ada Boost]
Test Accuracy: 0.8248
Train CV Accuracy: 0.803 (+/- 0.036) [Bagging]
Test Accuracy: 0.8167
Train CV Accuracy: 0.800 (+/- 0.034) [Gradient Boosting]
Test Accuracy: 0.8194
Train CV Accuracy: 0.792 (+/- 0.035) [XGBoost]
Test Accuracy: 0.7844
CPU times: user 8.74 s, sys: 1.25 s, total: 9.99 s
Wall time: 3.32 s
```

Tools that we used to solve this problem:

JUPYTER: We use Jupyter to load our dataset, and train and test the data using machine learning algorithms to predict the accuracy.

Numpy, Scikit-Learn, Pandas, Matplotlib,

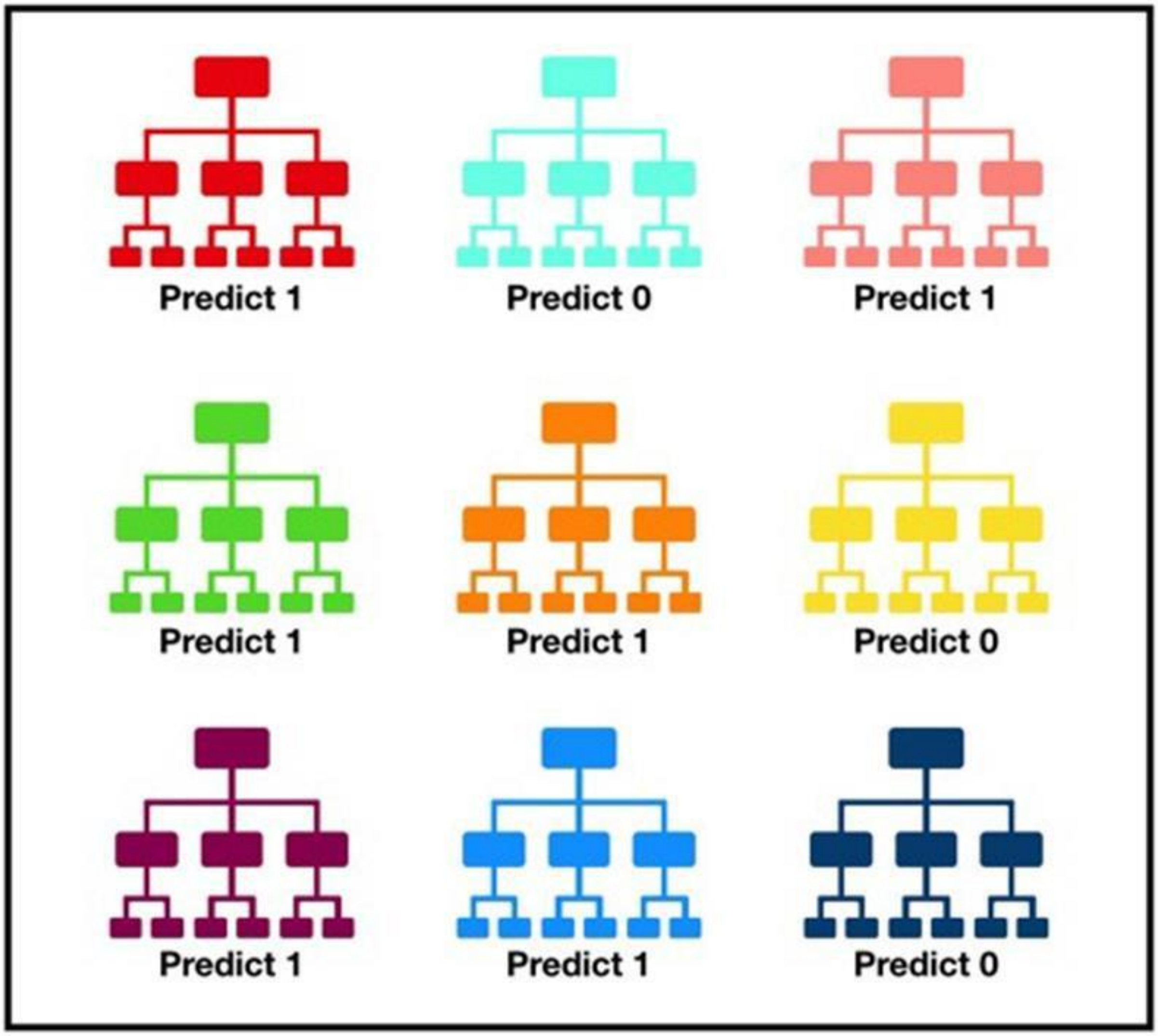
AWS : We use AWS for hosting the web application into a flask server.

Models that have been used to solve this problem :

We have used Random Forest Classifier,Gradient Boost and AdaBoost Classifier as they give higher accuracy than other models since they are combinations of different models.

The Random Forest Classifier

Like its name suggests, a random forest is made up of numerous independent decision trees that work together as an ensemble. Every every tree in the random forest spits out a class forecast, and the classification that receives the most votes becomes the prediction made by our model (see figure below).



Tally: Six 1s and Three 0s
Prediction: 1

The key is the low correlation between models. Uncorrelated models have the ability to provide ensemble forecasts that are more accurate than any of the individual predictions, just like assets with low correlations (like stocks and bonds) combine to form a portfolio that is greater than the sum of its parts. As long as they don't consistently all err in the same direction, the trees shield each other from their individual errors, which accounts for this lovely result. Many trees will be right while some may be wrong, allowing the group of trees to move in the proper direction. The following conditions must be met for random forest to function effectively

Gradient Boost

Three factors are involved in gradient boosting

A loss function that must be improved.

A poor predictor of the future.

A model that adds weak learners in order to reduce the loss function

1. Function Loss

The type of problem being solved determines the loss function to be employed. Although several common loss functions are available and you can create your own, it must be differentiable.For instance, classification and regression both sometimes employ logarithmic loss and squared error, respectively.The gradient boosting framework has the advantage that any differentiable loss function can be utilized, hence no new boosting algorithm needs to be developed for each loss function that could be desired to be employed.

2. A slow learner

In gradient boosting, decision trees are utilized as the weak learner.

Regression trees that produce real values for splits and whose output may be added allow for the "correction" of residuals in forecasts by adding the outputs of succeeding models.In building trees, greed is used to select the optimum split points based on purity scores like Gini or to reduce loss.At first, decision stumps—very short decision trees with just one split—were employed, as in the case of AdaBoost. Larger trees can typically be employed with four to eight levels.It is standard practice to impose restrictions on the weaker learners, such as a cap on the number of layers, nodes, splits, or leaf nodes.

3.Additive Model

One tree is added at a time, and the model's already-existing trees are left alone.When adding trees, a gradient descent approach is utilized to reduce loss.

Gradient descent is typically used to reduce a set of parameters, such as the weights in a neural network or the coefficients in a regression equation. The weights are changed to reduce inaccuracy after calculating loss or error.

Weak learner sub-models, or more particularly decision trees, are used in place of parameters. We must incorporate a tree into the model to lower the loss before we can begin the gradient descent technique (i.e. follow the gradient). To achieve this, we parameterize the tree, change its parameters, and then move the tree.

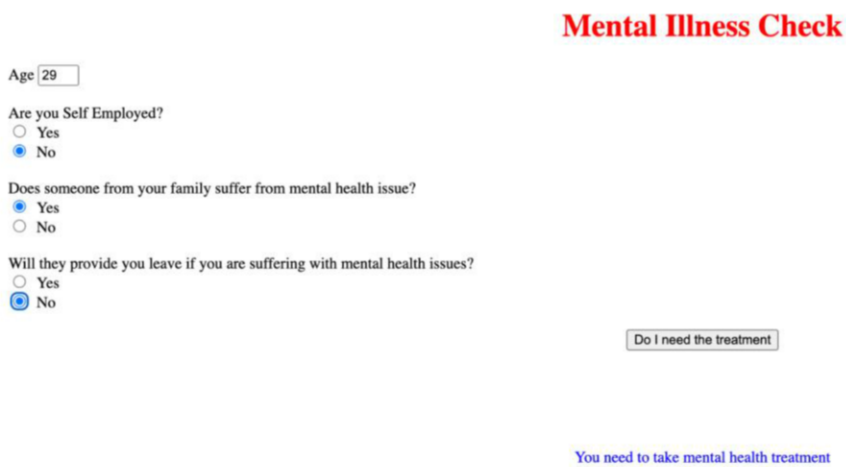
Ada Boost

A highly well-liked boosting technique called AdaBoost (Adaptive Boosting) seeks to combine several weak classifiers into one powerful classifier. Yoav Freund and Robert Schapire wrote the first AdaBoost paper.

A single classifier might not be able to predict an object's class with sufficient accuracy, but by grouping several weak classifiers and having each one gradually learn from the incorrectly classified items of the others, we can create one such strong model. The classifier given here could be any of your standard classifiers, including Logistic Regression, Decision Trees (which are frequently the default), and others.

Results Section

By using flask we created an application which predicts whether an employee needs to take treatment or not



Mental Illness Check

Age 29

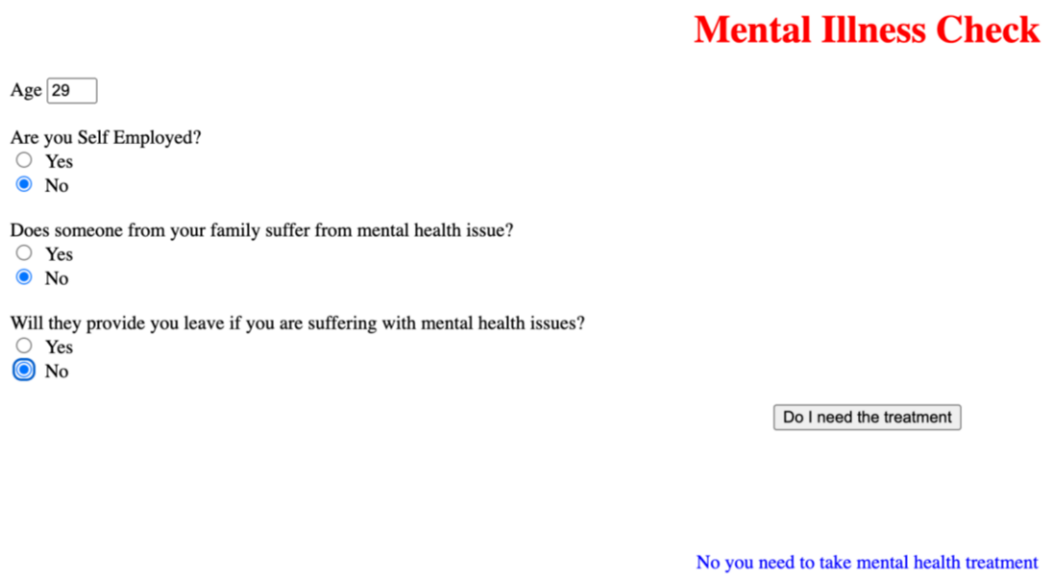
Are you Self Employed?
- ○ Yes
- ◉ No

Does someone from your family suffer from mental health issue?
- ◉ Yes
- ○ No

Will they provide you leave if you are suffering with mental health issues?
- ○ Yes
- ◉ No

Do I need the treatment

You need to take mental health treatment



Mental Illness Check

Age 29

Are you Self Employed?
- ○ Yes
- ◉ No

Does someone from your family suffer from mental health issue?
- ○ Yes
- ◉ No

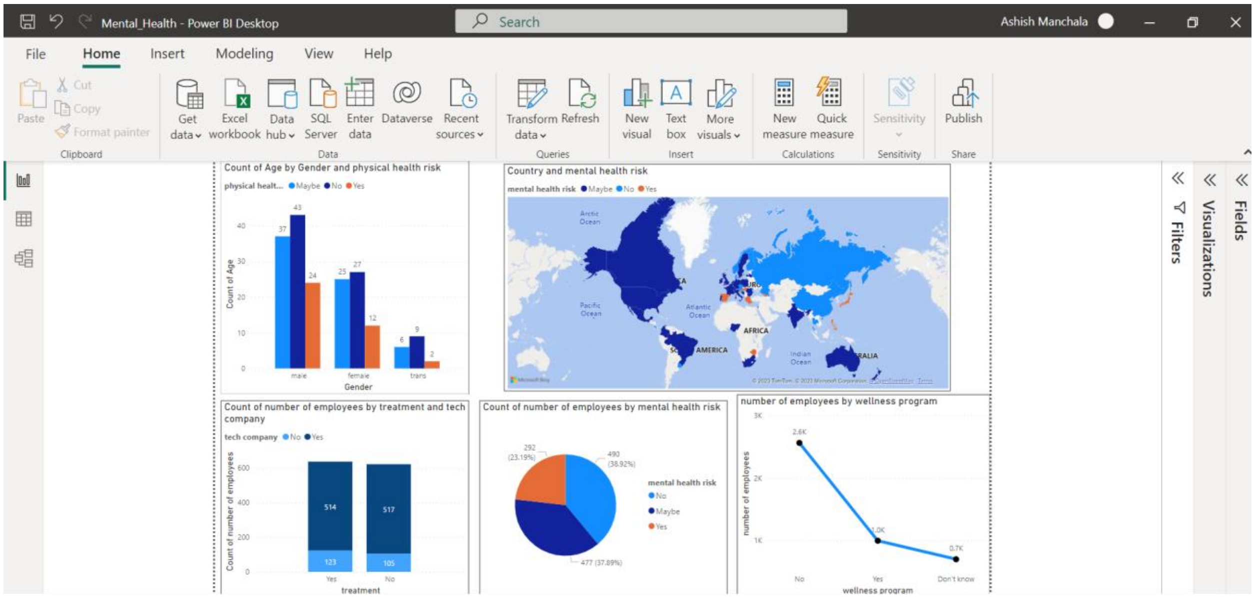
Will they provide you leave if you are suffering with mental health issues?
- ○ Yes
- ◉ No

Do I need the treatment

No you need to take mental health treatment

Visualizations Using Power BI:

We have created visualizations using Power BI from the dataset.



References

https://www.researchgate.net/publication/268223440_Mental_health-related_stigma_in_health_care_and_mental_health-care

https://www.researchgate.net/publication/264502507_Employee_Health_in_the_Mental_Health_Workplace_Clinical_Administrative_and_Organizational_Perspectiv es

https://www.kaggle.com/code/aditimulye/mental-health-at-workplace