Mental Health Prediction Using ML

Made by-Ankur Rohilla, Arnav Chhikara, Dhruv Shah

Project Overview:

Mental Health First Aid teaches participants how to notice and support an individual who may be experiencing a mental health or substance use concern or crisis and connect them with the appropriate employee resources.

Employers can offer robust benefit packages to support employees who go through mental health issues. That includes Employee Assistance Programs, Wellness programs that focus on mental and physical health, Health and Disability Insurance or flexible working schedules or time off policies. Organisations that incorporate mental health awareness help to create a healthy and productive work environment that reduces the stigma associated with mental illness, increases the organisations mental health literacy and teaches the skills to safely and responsibly respond to a co-workers mental health concern.

The main purpose of the Mental Health Prediction system is to predict whether a person needs to seek Mental health treatment or not based on inputs provided by them.

We will be using classification algorithms such as Logistic Regression, KNN, Decision tree, Random forest, AdaBoost, GradientBoost and XGBoost. We will train and test the data with these algorithms. From this the best model is selected and saved in pkl format. We will also be deploying our model locally using Flask.

Purpose:

By the end of this project we will:

- Know fundamental concepts and techniques used for machine learning.
- Gain a broad understanding about data.
- Have knowledge on pre-processing the data/transformation techniques and some visualisation concepts before building the model
- Learn how to build a machine learning model and tune it for better performance
- Know how to evaluate the model and deploy it using flask

Existing Problem:

Predicting mental health using machine learning

References:

To complete this project, you will require the following softwares, concepts and packages

- **Anaconda navigator:**
  - Refer the link below to download anaconda navigator o Link : https://youtu.be/1ra4zH2G4o0
- **Python packages:**
  - Open anaconda prompt as administrator ○ Type “pip install numpy” and click enter. ○ Type “pip install pandas” and click enter. ○ Type “pip install scikit-learn” and click enter. ○ Type ”pip install matplotlib” and click enter.
  - Type ”pip install scipy” and click enter.
  - Type ”pip install pickle-mixin” and click enter. ○ Type ”pip install seaborn” and click enter. ○ Type “pip install Flask” and click enter.

- **ML Concepts**
  - Supervised learning: https://www.javatpoint.com/supervised-machine-learning ○ Unsupervised learning: https://www.javatpoint.com/unsupervised-machine-learning

  - Regression and classification ▪ Logistic regression:

    https://www.javatpoint.com/logistic-regression-in-machine-learning
    - Decision tree: https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm
    - Random forest: https://www.javatpoint.com/machine-learning-random-forest-algorithm
    - KNN: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning
    - Xgboost:

https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/

- AdaBoost: https://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-complete-guide-for-beginners/

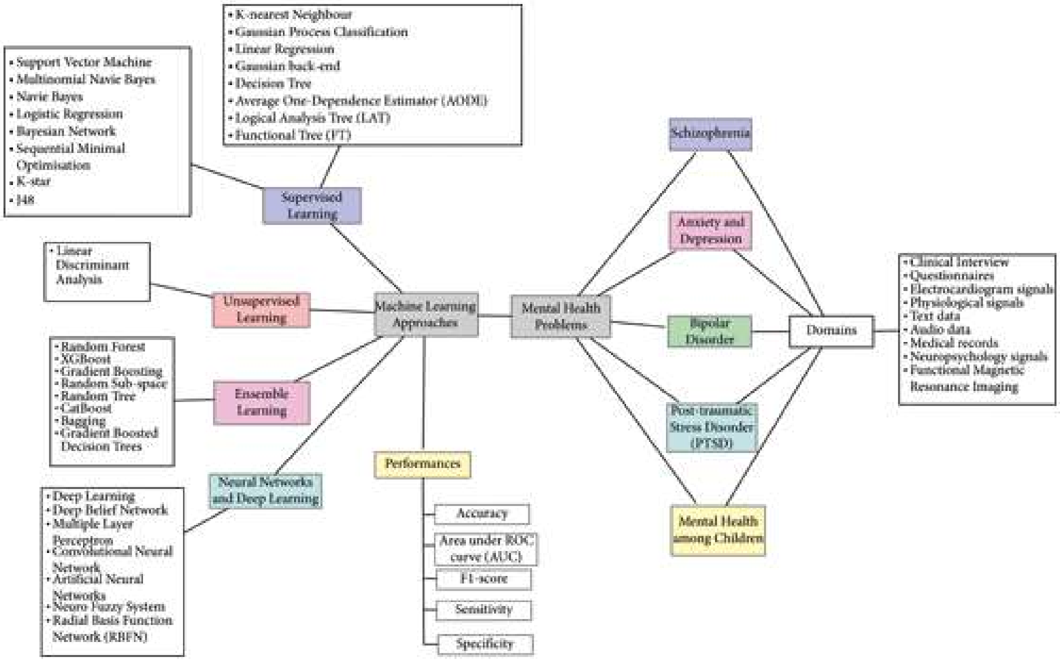
- Gradient Boost: https://www.analyticsvidhya.com/blog/2021/09/gradient-boosting-algorithm-a-complete-guide-for-beginners/
- Evaluation metrics: https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/

- **Flask Basics** : https://www.youtube.com/watch?v=lj4I_CvBnt0

Problem statement definition:

The main purpose of the Mental Health Prediction system is to predict whether a person needs to seek Mental health treatment or not based on inputs provided by them.

Empathy map canvas:



Ideation and Brainstorming:

The Says quadrant contains what the user says out loud in an interview or some other usability study. Ideally, it contains verbatim and direct quotes from research.

- I want to do this but I can't do this alone
- I have to set my limits □ I'm motivated

The Thinks quadrant captures what the user is thinking throughout the experience. Ask yourself (from the qualitative research gathered): what occupies the user's thoughts? What matters to the user? It is possible to have the same content in both Says and Thinks. However, pay special attention to what users think, but may not be willing to vocalize. Try to understand why they are reluctant to share — are they unsure, selfconscious, polite, or afraid to tell others something?

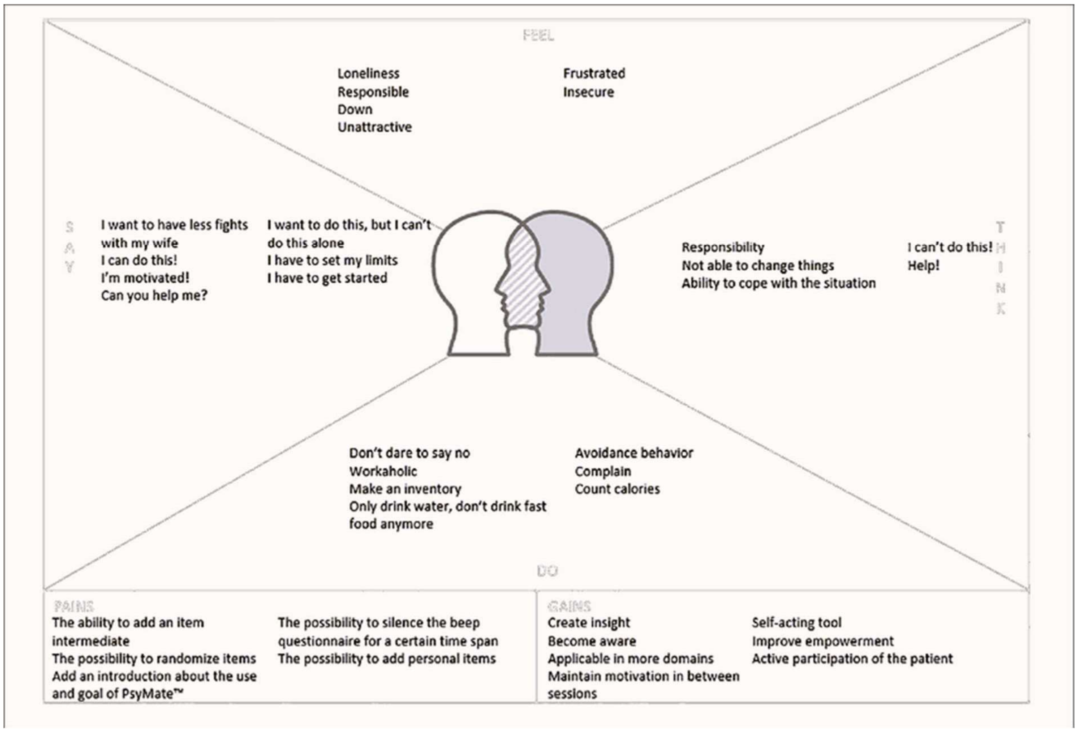
- Help
- Responsibility

The Does quadrant encloses the actions the user takes. From the research, what does the user physically do? How does the user go about doing it?

- Make an inventory
- Avoidance behaviour
- Complain

The Feels quadrant is the user's emotional state, often represented as an adjective plus a short sentence for context. Ask yourself: what worries the user? What does the user get excited about? How does the user feel about the experience?

- Loneliness
- Insecure
- Frustrated


FEEL
Loneliness
Responsible
Down
Unattractive
Frustrated
Insecure
SAY
I want to have less fights with my wife
I can do this!
I'm motivated!
Can you help me?
I want to do this, but I can't do this alone
I have to set my limits
I have to get started
THINK
Responsibility
Not able to change things
Ability to cope with the situation
I can't do this!
Help!
DO
Don't dare to say no
Workaholic
Make an inventory
Only drink water, don't drink fast food anymore
Avoidance behavior
Complain
Count calories
PAINS
The ability to add an item intermediate
The possibility to randomize items
Add an introduction about the use and goal of PsyMate™
The possibility to silence the beep questionnaire for a certain time span
The possibility to add personal items
GAINS
Create insight
Become aware
Applicable in more domains
Maintain motivation in between sessions
Self-acting tool
Improve empowerment
Active participation of the patient

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate,
helping each other develop a rich amount of creative solutions.
Reference: https://www.mural.co/templates/empathy-map-canvas

Requirements:

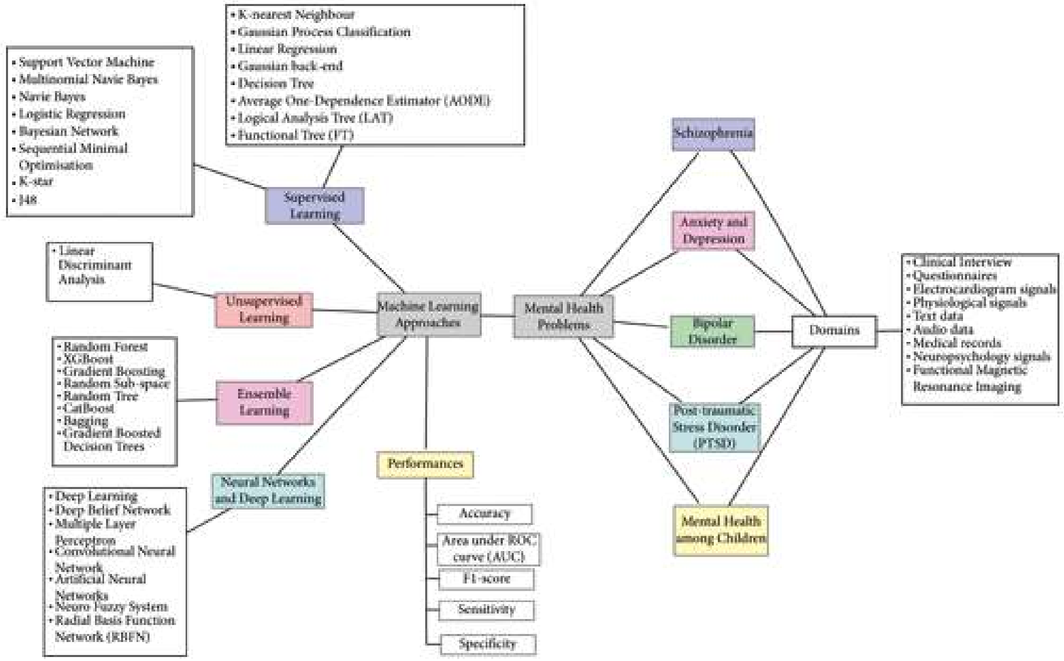
- **ML Concepts**
  - Supervised learning: https://www.javatpoint.com/supervised-machine-learning o Unsupervised learning: https://www.javatpoint.com/unsupervised-machine-learning
  - Regression and classification ▪ Logistic regression:
    https://www.javatpoint.com/logistic-regression-in-machine-learning
    - Decision tree: https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm

- Random forest: https://www.javatpoint.com/machine-learning-random-forest-algorithm
- KNN: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning
- Xgboost: https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/
- AdaBoost: https://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-complete-guide-for-beginners/
- Gradient Boost: https://www.analyticsvidhya.com/blog/2021/09/gradient-boosting-algorithm-a-complete-guide-for-beginners/
- Evaluation metrics: https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/

- **Flask Basics** : https://www.youtube.com/watch?v=lj4I_CvBnt0

Project Design:

Data flow diagram:-



Solution architecture :

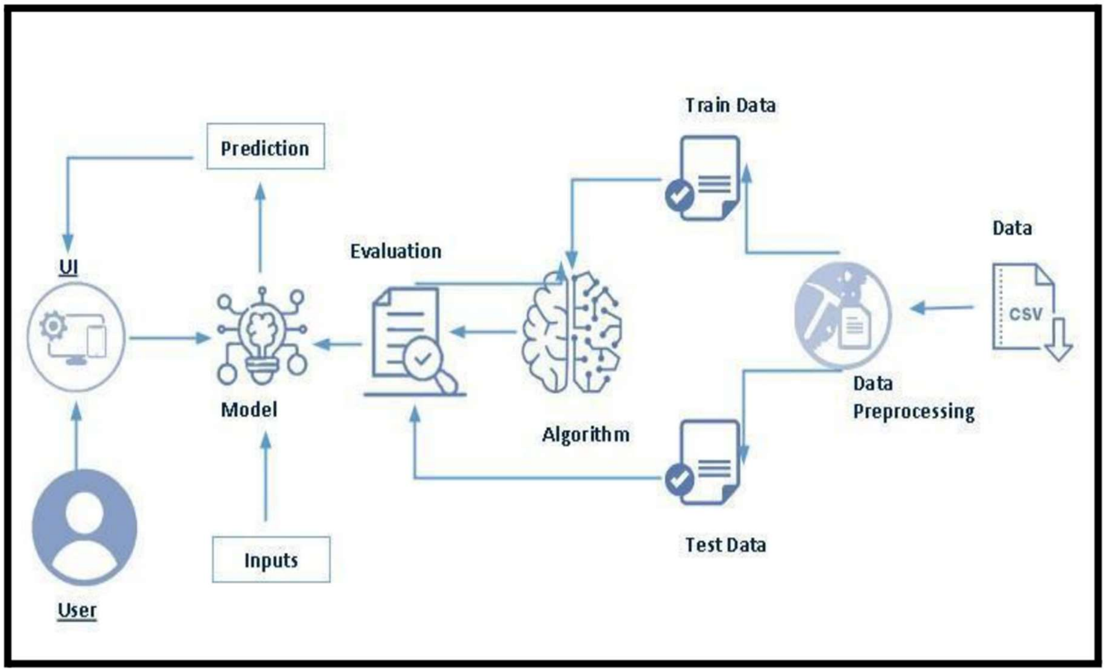
- User interacts with the UI to enter the input.
- Entered input is analysed by the model which is integrated.
- The predictions made by the model is showcased on the UI

To accomplish this, we have to complete all the activities listed below,

- Data collection ○ Collect the dataset or create the dataset
- Data pre-processing ○ Removing unnecessary columns ○ Checking for null values
- Visualising and analysing data ○ Univariate analysis ○ Bivariate analysis ○ Descriptive analysis
- Model building ○ Handling categorical values ○ Dividing data into train and test sets ○ Import the model building libraries ○ Comparing accuracy of various models ○ Hyperparameter tuning of the selected model ○ Evaluating performance of models ○ Save the model
- Application Building ○ Create an HTML file ○ Build python code

PROJECT PLANNING & SCHEDULING:

Technical Architecture:

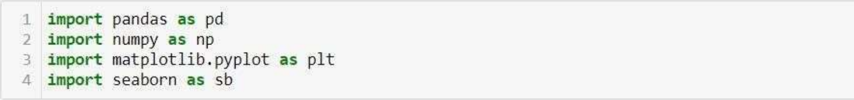


Sprint planning and estimation:

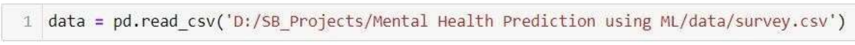
- User interacts with the UI to enter the input.
- Entered input is analysed by the model which is integrated.
- The predictions made by the model is showcased on the UI

To accomplish this, we have to complete all the activities listed below,

- Data collection ○ Collect the dataset or create the dataset
- Data pre-processing ○ Removing unnecessary columns ○ Checking for null values
- Visualising and analysing data ○ Univariate analysis ○ Bivariate analysis ○ Descriptive analysis
- Model building ○ Handling categorical values ○ Dividing data into train and test sets ○ Import the model building libraries ○ Comparing accuracy of various models ○ Hyperparameter tuning of the selected model ○ Evaluating performance of models ○ Save the model
- Application Building ○ Create an HTML file ○ Build python code



```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sb
```



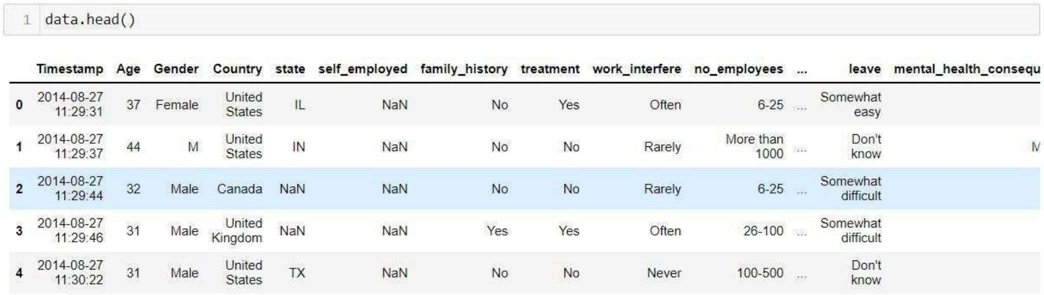
```
data = pd.read_csv('D:/SB_Projects/Mental Health Prediction using ML/data/survey.csv')
```

- h.ipynb
- model.pkl is our saved model. We will use this model for flask integration.

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

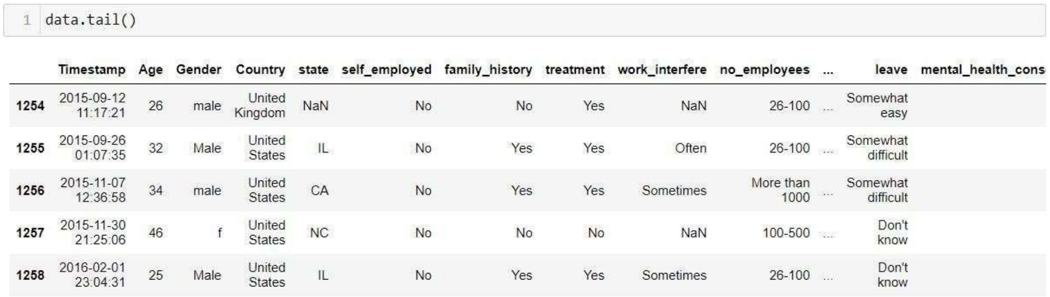
Load the dataset using read_csv() function:

Inside the read_csv() function, specify the path to your dataset.



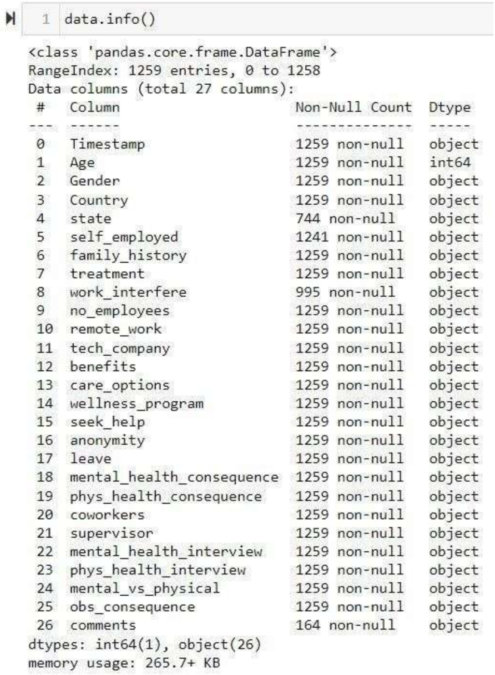
```
data.head()
```

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



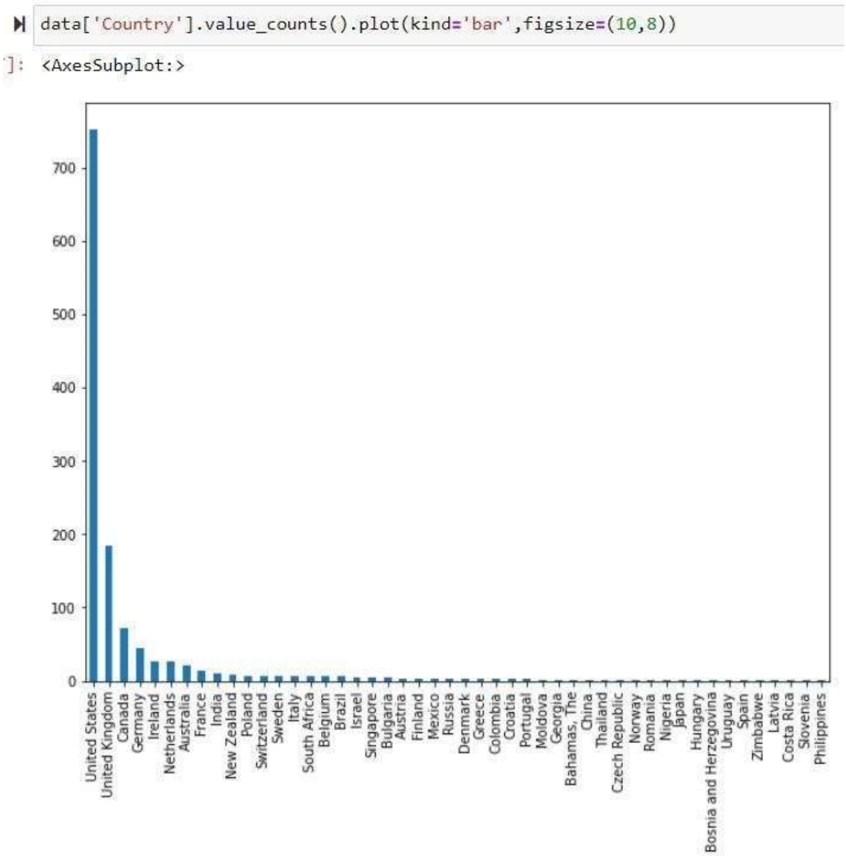
```
data.tail()
```

	Timestamp	Age	Gender	Country	state	self_employed	family_history	treatment	work_interfere	no_employees	...	leave	mental_health_cons
1254	2015-09-12 11:17:21	26	male	United Kingdom	NaN	No	No	Yes	NaN	26-100	...	Somewhat easy	
1255	2015-09-26 01:07:35	32	Male	United States	IL	No	Yes	Yes	Often	26-100	...	Somewhat difficult	
1256	2015-11-07 12:36:58	34	male	United States	CA	No	Yes	Yes	Sometimes	More than 1000	...	Somewhat difficult	
1257	2015-11-30 21:25:06	46	f	United States	NC	No	No	No	NaN	100-500	...	Don't know	
1258	2016-02-01 23:04:31	25	Male	United States	IL	No	Yes	Yes	Sometimes	26-100	...	Don't know	



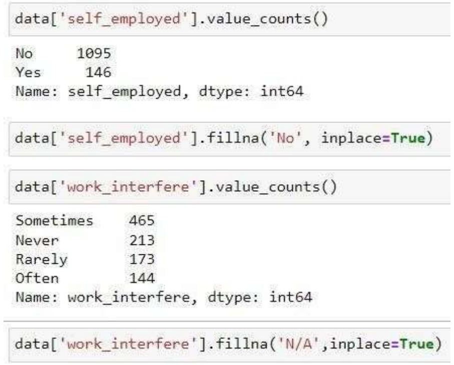
```
data.info()
```

```
<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
```



```
data['Country'].value_counts().plot(kind='bar',figsize=(10,8))
```

```
]: <AxesSubplot:>
```



```
data['self_employed'].value_counts()
```

```
No     1095
Yes     146
Name: self_employed, dtype: int64
```

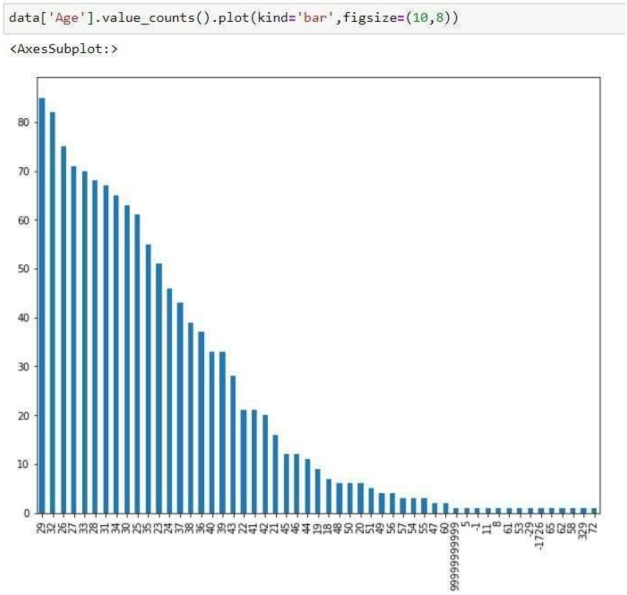
```
data['self_employed'].fillna('No', inplace=True)
```

```
data['work_interfere'].value_counts()
```

```
Sometimes    465
Never        213
Rarely       173
Often        144
Name: work_interfere, dtype: int64
```

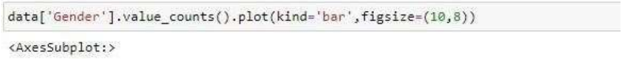
```
data['work_interfere'].fillna('N/A',inplace=True)
```

Now our dataset is free of null values.



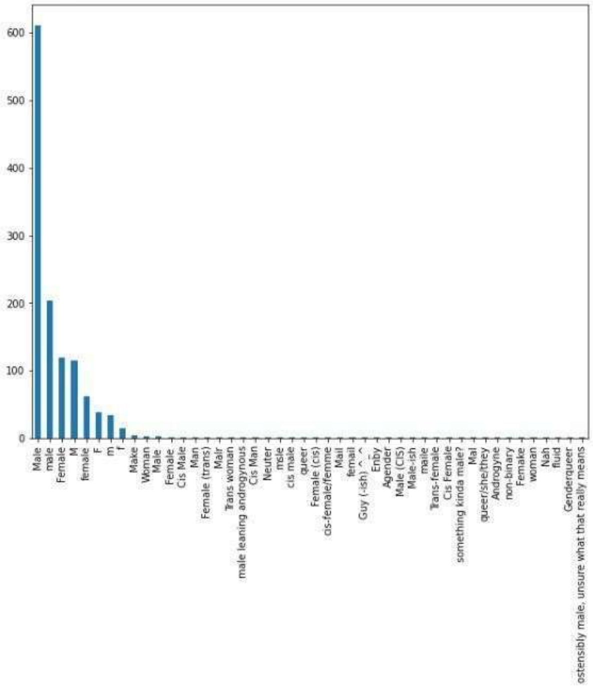
```
data['Age'].value_counts().plot(kind='bar',figsize=(10,8))
```

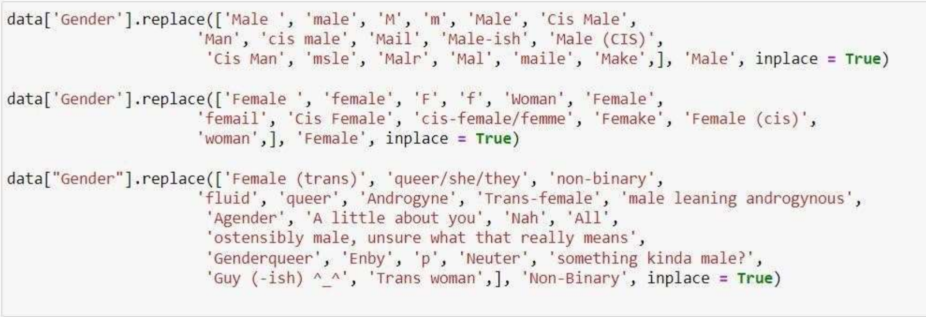
```
<AxesSubplot:>
```



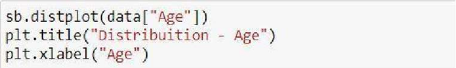
```
data['Gender'].value_counts().plot(kind='bar',figsize=(10,8))
```

```
<AxesSubplot:>
```

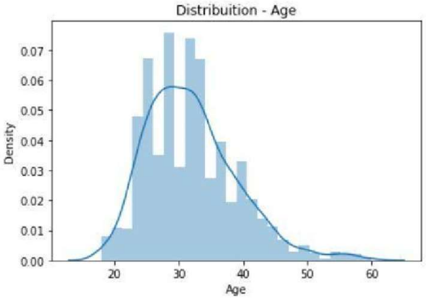




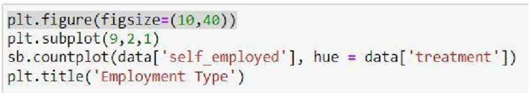
```
data['Gender'].replace(['Male ', 'male', 'M', 'm', 'Male', 'Cis Male',
                        'Man', 'cis male', 'Mail', 'Male-ish', 'Male (CIS)',
                         'Cis Man', 'msle', 'Malr', 'Mal', 'maile', 'Make',], 'Male', inplace = True)

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

data["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',], 'Non-Binary', inplace = True)
```



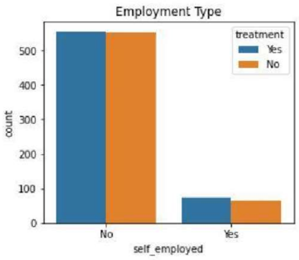
```
sb.distplot(data["Age"])
plt.title("Distribuition - Age")
plt.xlabel("Age")
```



- Employment type and treatment

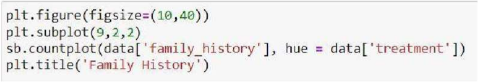


```
plt.figure(figsize=(10,40))
plt.subplot(9,2,1)
sb.countplot(data['self_employed'], hue = data['treatment'])
plt.title('Employment Type')
```

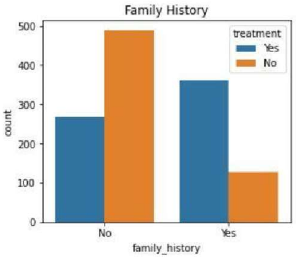


We observe that though there is a vast difference between people who are self employed or not, the number of people who seek treatment in both the categories is more or less similar.

- Family history and treatment

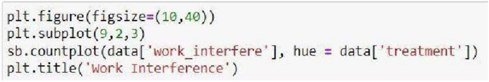


```
plt.figure(figsize=(10,40))
plt.subplot(9,2,2)
sb.countplot(data['family_history'], hue = data['treatment'])
plt.title('Family History')
```

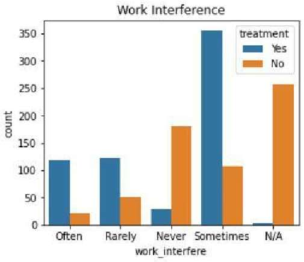


We observe that treatment is directly proportional to family history. Hence this is an important factor.

- Work interference and treatment

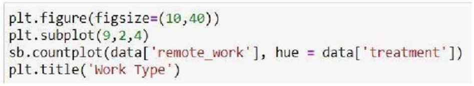


```
plt.figure(figsize=(10,40))
plt.subplot(9,2,3)
sb.countplot(data['work_interfere'], hue = data['treatment'])
plt.title('Work Interference')
```

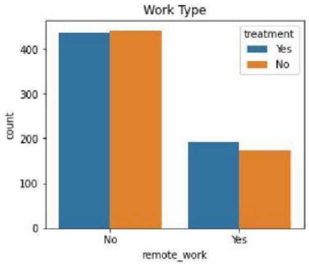


We observe that the people who chose Sometimes were the largest who wanted to get treatment. These group of people are the ones who are reluctant to choose either of the extreme categories.

- Work type and treatment

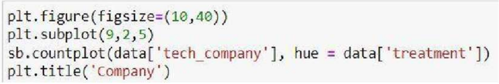


```
plt.figure(figsize=(10,40))
plt.subplot(9,2,4)
sb.countplot(data['remote_work'], hue = data['treatment'])
plt.title('Work Type')
```

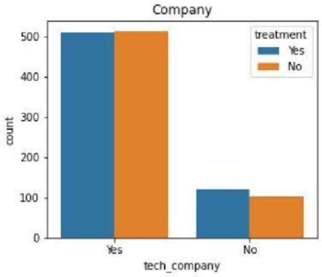


We observe that the number of people who seek treatment in both the categories is more or less similar and it does not affect our target variable.

- Company and treatment

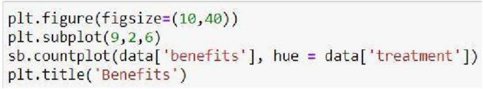


```
plt.figure(figsize=(10,40))
plt.subplot(9,2,5)
sb.countplot(data['tech_company'], hue = data['treatment'])
plt.title('Company')
```

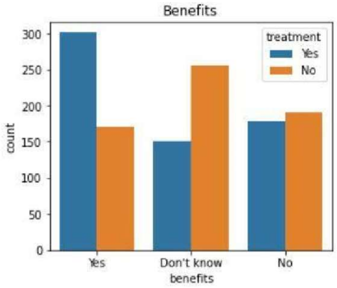


We can conclude that irrespective of the field the company of the people falls in, mental health is a big issue.

- Benefits and treatment

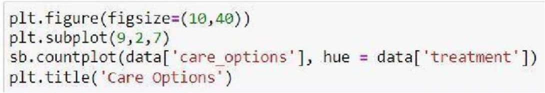


```
plt.figure(figsize=(10,40))
plt.subplot(9,2,6)
sb.countplot(data['benefits'], hue = data['treatment'])
plt.title('Benefits')
```

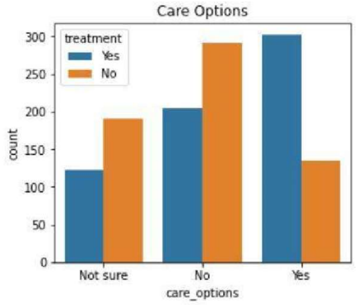


We see that a large group among the people who wanted mental health benefits wanted to seek treatment and also a significant number of people who said No too, wanted to seek treatment.

- Care options and treatment

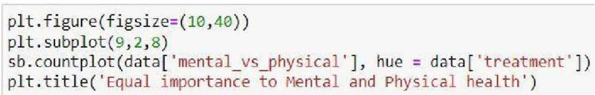


```
plt.figure(figsize=(10,40))
plt.subplot(9,2,7)
sb.countplot(data['care_options'], hue = data['treatment'])
plt.title('Care Options')
```

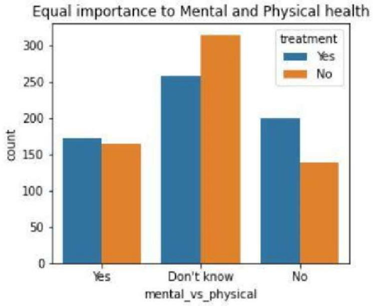


This graph is quite similar to the benefits column.

- Mental vs Physical health

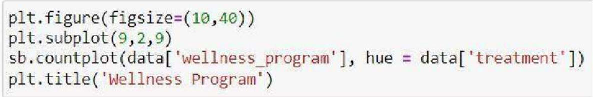


```
plt.figure(figsize=(10,40))
plt.subplot(9,2,8)
sb.countplot(data['mental_vs_physical'], hue = data['treatment'])
plt.title('Equal importance to Mental and Physical health')
```

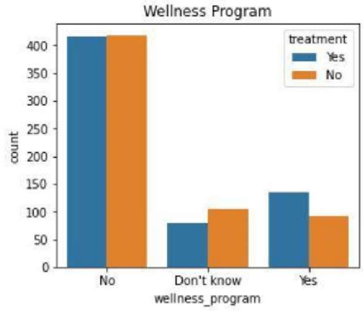


We observe that half of the people are not aware of the importance given to mental health as compared to physical health, whereas almost equal parts of the other halves answered Yes and No.

- Wellness program and treatment

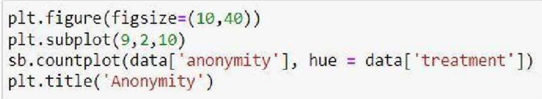


```
plt.figure(figsize=(10,40))
plt.subplot(9,2,9)
sb.countplot(data['wellness_program'], hue = data['treatment'])
plt.title('Wellness Program')
```

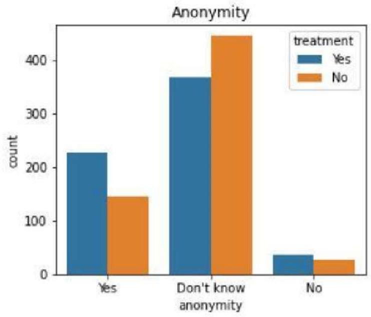


We observe that almost half of the people who said No want to seek treatment, which means their company has to take steps in arranging for the same.

- Anonymity and treatment

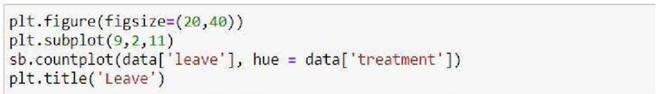


```
plt.figure(figsize=(10,40))
plt.subplot(9,2,10)
sb.countplot(data['anonymity'], hue = data['treatment'])
plt.title('Anonymity')
```

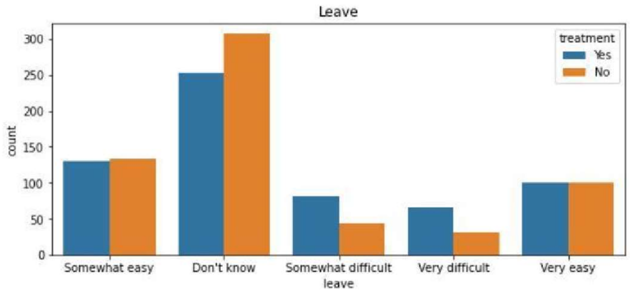


We observe that most people either answered yes or they are not aware if their anonymity will be protected.

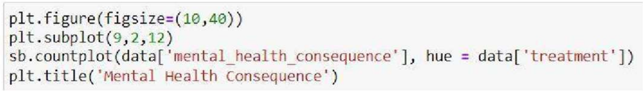
- Leave and treatment



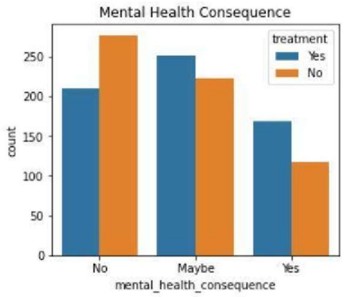
```
plt.figure(figsize=(20,40))
plt.subplot(9,2,11)
sb.countplot(data['leave'], hue = data['treatment'])
plt.title('Leave')
```



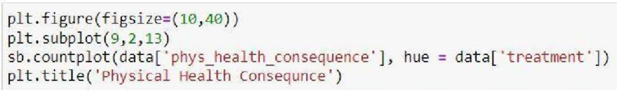
- Mental health consequence and treatment



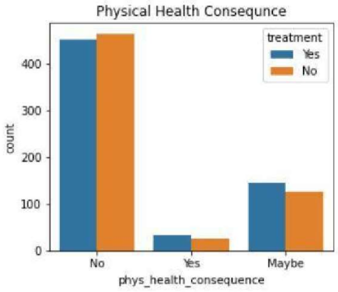
```
plt.figure(figsize=(10,40))
plt.subplot(9,2,12)
sb.countplot(data['mental_health_consequence'], hue = data['treatment'])
plt.title('Mental Health Consequence')
```



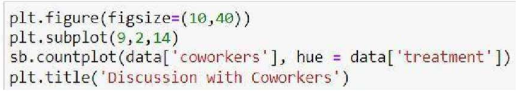
- Physical health consequence and treatment



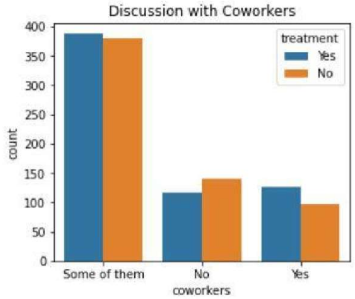
```
plt.figure(figsize=(10,40))
plt.subplot(9,2,13)
sb.countplot(data['phys_health_consequence'], hue = data['treatment'])
plt.title('Physical Health Consequnce')
```



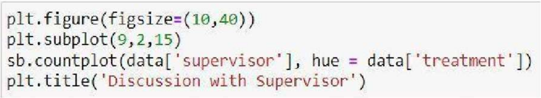
- Discussion with coworkers and treatment



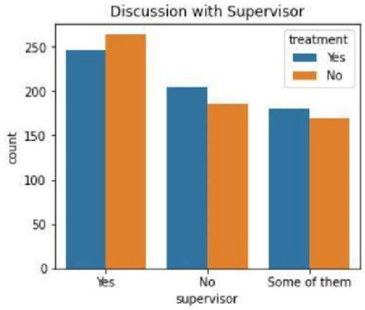
```
plt.figure(figsize=(10,40))
plt.subplot(9,2,14)
sb.countplot(data['coworkers'], hue = data['treatment'])
plt.title('Discussion with Coworkers')
```



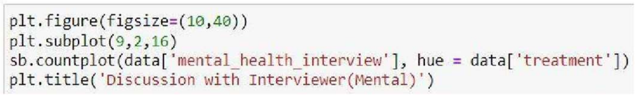
- Discussion with supervisor and treatment



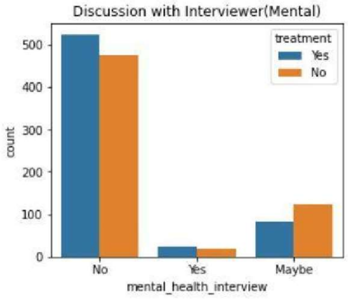
```
plt.figure(figsize=(10,40))
plt.subplot(9,2,15)
sb.countplot(data['supervisor'], hue = data['treatment'])
plt.title('Discussion with Supervisor')
```



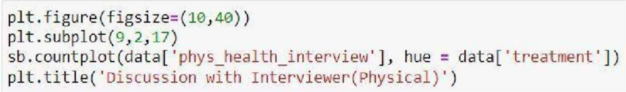
- Mental health discussion during interview and treatment



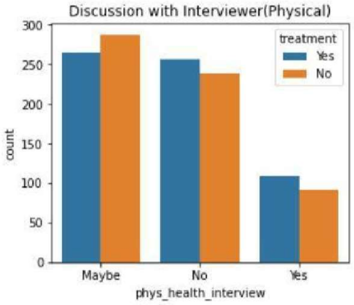
```
plt.figure(figsize=(10,40))
plt.subplot(9,2,16)
sb.countplot(data['mental_health_interview'], hue = data['treatment'])
plt.title('Discussion with Interviewer(Mental)')
```



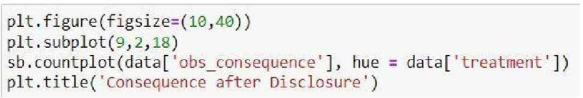
- Physical health discussion during interview and treatment



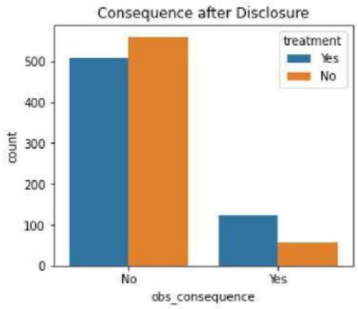
```
plt.figure(figsize=(10,40))
plt.subplot(9,2,17)
sb.countplot(data['phys_health_interview'], hue = data['treatment'])
plt.title('Discussion with Interviewer(Physical)')
```

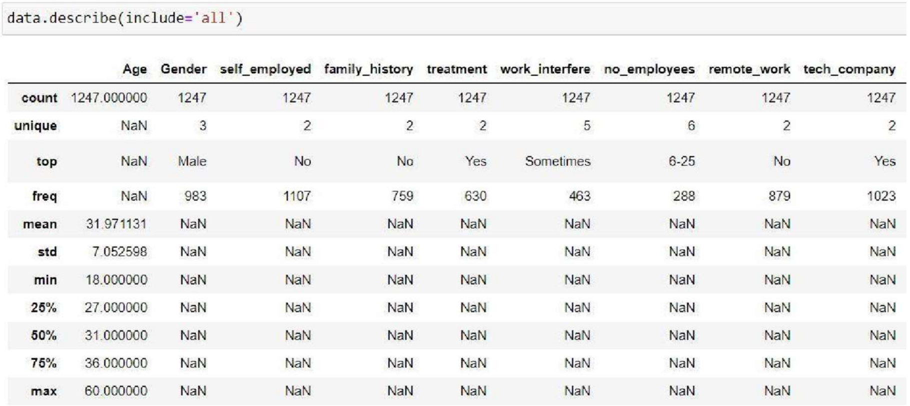


- Consequence after disclosure and treatment



```
plt.figure(figsize=(10,40))
plt.subplot(9,2,18)
sb.countplot(data['obs_consequence'], hue = data['treatment'])
plt.title('Consequence after Disclosure')
```

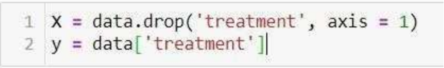




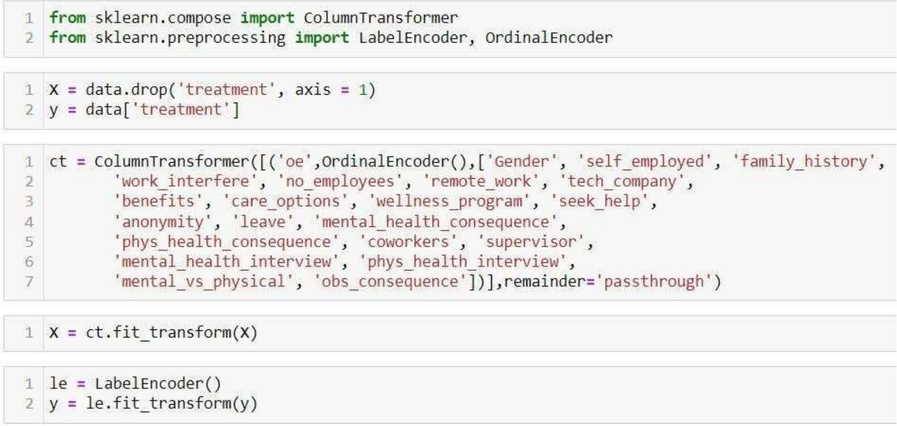
```
data.describe(include='all')
```

	Age	Gender	self_employed	family_history	treatment	work_interfere	no_employees	remote_work	tech_company
count	1247.000000	1247	1247	1247	1247	1247	1247	1247	1247
unique	NaN	3	2	2	2	5	6	2	2
top	NaN	Male	No	No	Yes	Sometimes	6-25	No	Yes
freq	NaN	983	1107	759	630	463	288	879	1023
mean	31.971131	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
std	7.052598	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
min	18.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
25%	27.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
50%	31.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
75%	36.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
max	60.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Performance matrix:



```
X = data.drop('treatment', axis = 1)
y = data['treatment']
```



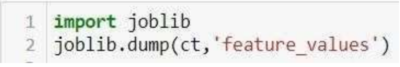
```
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import LabelEncoder, OrdinalEncoder
```

```
X = data.drop('treatment', axis = 1)
y = data['treatment']
```

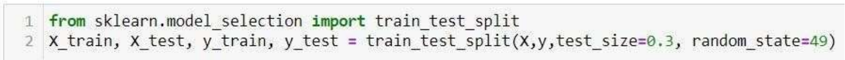
```
ct = ColumnTransformer([('oe',OrdinalEncoder(),['Gender', 'self_employed', 'family_history',
        'work_interfere', 'no_employees', 'remote_work', 'tech_company',
        'benefits', 'care_options', 'wellness_program', 'seek_help',
        'anonymity', 'leave', 'mental_health_consequence',
        'phys_health_consequence', 'coworkers', 'supervisor',
        'mental_health_interview', 'phys_health_interview',
        'mental_vs_physical', 'obs_consequence'])],remainder='passthrough')
```

```
X = ct.fit_transform(X)
```

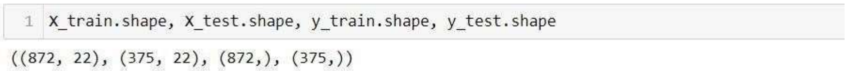
```
le = LabelEncoder()
y = le.fit_transform(y)
```



```
import joblib
joblib.dump(ct,'feature_values')
```

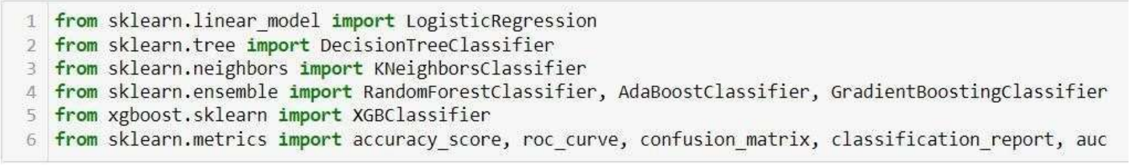


```python
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3, random_state=49)
```

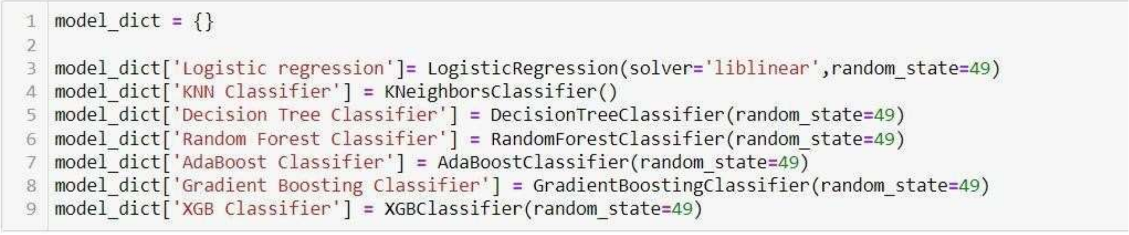


```python
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

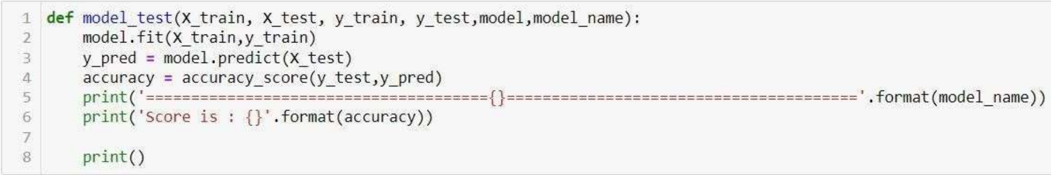
```
((872, 22), (375, 22), (872,), (375,))
```



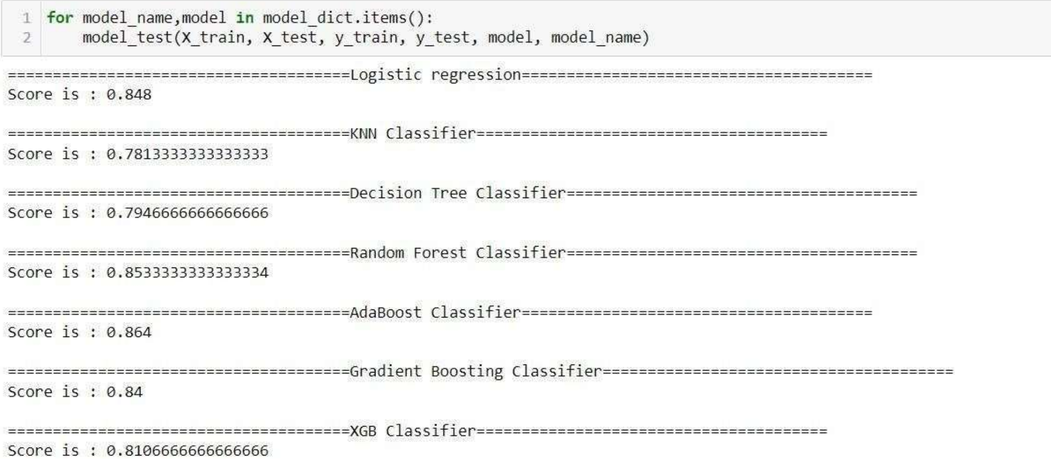
```python
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier, GradientBoostingClassifier
from xgboost.sklearn import XGBClassifier
from sklearn.metrics import accuracy_score, roc_curve, confusion_matrix, classification_report, auc
```



```python
model_dict = {}

model_dict['Logistic regression']= LogisticRegression(solver='liblinear',random_state=49)
model_dict['KNN Classifier'] = KNeighborsClassifier()
model_dict['Decision Tree Classifier'] = DecisionTreeClassifier(random_state=49)
model_dict['Random Forest Classifier'] = RandomForestClassifier(random_state=49)
model_dict['AdaBoost Classifier'] = AdaBoostClassifier(random_state=49)
model_dict['Gradient Boosting Classifier'] = GradientBoostingClassifier(random_state=49)
model_dict['XGB Classifier'] = XGBClassifier(random_state=49)
```

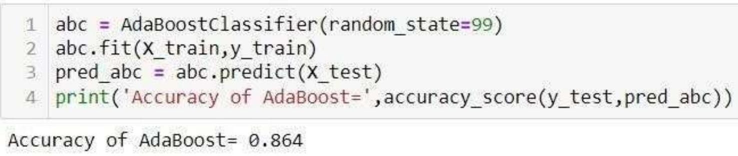


```python
def model_test(X_train, X_test, y_train, y_test,model,model_name):
    model.fit(X_train,y_train)
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test,y_pred)
    print('======================================{}======================================='.format(model_name))
    print('Score is : {}'.format(accuracy))

    print()
```



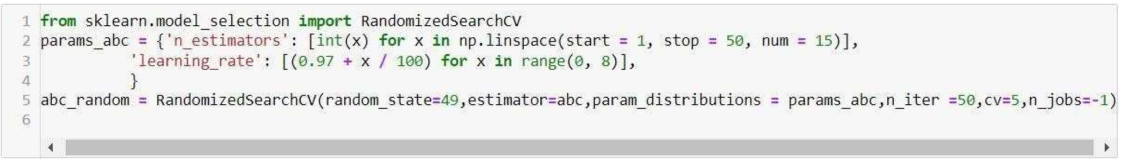
```python
for model_name,model in model_dict.items():
    model_test(X_train, X_test, y_train, y_test, model, model_name)
```

```
======================================Logistic regression=======================================
Score is : 0.848

======================================KNN Classifier=======================================
Score is : 0.7813333333333333

======================================Decision Tree Classifier=======================================
Score is : 0.7946666666666666

======================================Random Forest Classifier=======================================
Score is : 0.8533333333333334

======================================AdaBoost Classifier=======================================
Score is : 0.864

======================================Gradient Boosting Classifier=======================================
Score is : 0.84

======================================XGB Classifier=======================================
Score is : 0.8106666666666666
```

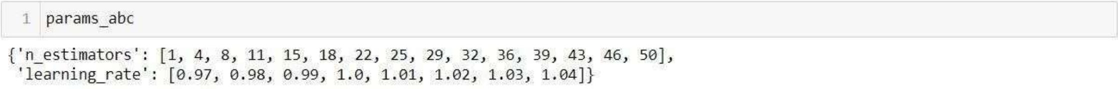


```python
abc = AdaBoostClassifier(random_state=99)
abc.fit(X_train,y_train)
pred_abc = abc.predict(X_test)
print('Accuracy of AdaBoost=',accuracy_score(y_test,pred_abc))
```

```
Accuracy of AdaBoost= 0.864
```



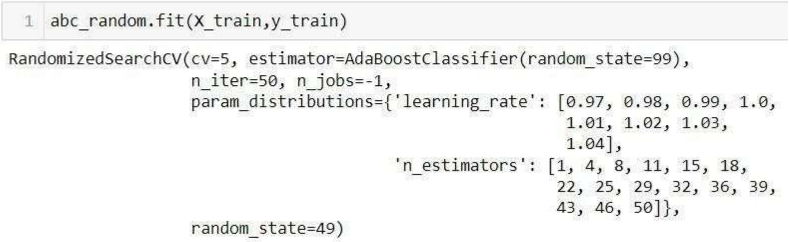
```
from sklearn.model_selection import RandomizedSearchCV
params_abc = {'n_estimators': [int(x) for x in np.linspace(start = 1, stop = 50, num = 15)],
          'learning_rate': [(0.97 + x / 100) for x in range(0, 8)],
          }
abc_random = RandomizedSearchCV(random_state=49,estimator=abc,param_distributions = params_abc,n_iter =50,cv=5,n_jobs=-1)

```



```
params_abc
```

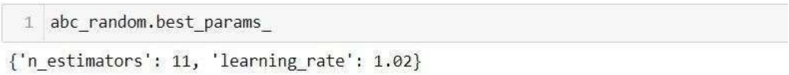
```
{'n_estimators': [1, 4, 8, 11, 15, 18, 22, 25, 29, 32, 36, 39, 43, 46, 50],
 'learning_rate': [0.97, 0.98, 0.99, 1.0, 1.01, 1.02, 1.03, 1.04]}
```

For n_estimators, we are taking 15 equally spaced values from 1 to 50 and for learning rate, we are trying various values close to 1.



```
abc_random.fit(X_train,y_train)
```

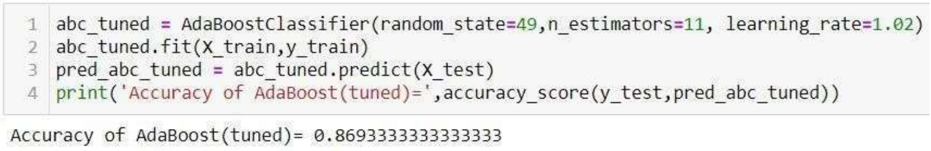
```
RandomizedSearchCV(cv=5, estimator=AdaBoostClassifier(random_state=99),
                   n_iter=50, n_jobs=-1,
                   param_distributions={'learning_rate': [0.97, 0.98, 0.99, 1.0,
                                                          1.01, 1.02, 1.03,
                                                          1.04],
                                        'n_estimators': [1, 4, 8, 11, 15, 18,
                                                         22, 25, 29, 32, 36, 39,
                                                         43, 46, 50]},
                   random_state=49)
```



```
abc_random.best_params_
```

```
{'n_estimators': 11, 'learning_rate': 1.02}
```

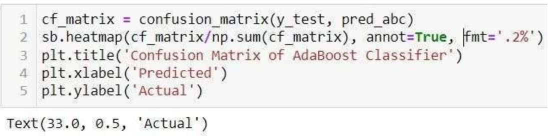
So, our model will perform the best if n_estimators are equal to 11 and learning_rate is equal to 1.02. Let us add these values to train our model, make predictions and check accuracy.



```
abc_tuned = AdaBoostClassifier(random_state=49,n_estimators=11, learning_rate=1.02)
abc_tuned.fit(X_train,y_train)
pred_abc_tuned = abc_tuned.predict(X_test)
print('Accuracy of AdaBoost(tuned)=',accuracy_score(y_test,pred_abc_tuned))
```

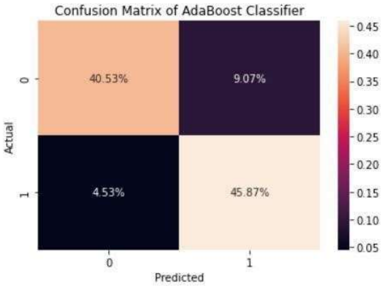
```
Accuracy of AdaBoost(tuned)= 0.8693333333333333
```

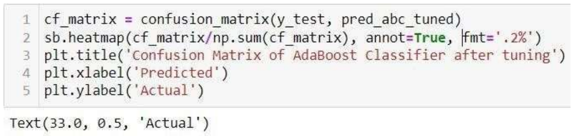
We observe that the accuracy has increased approximately by 0.5%. Though this is not a very great improvement, it is at least better than our previous model.



```
cf_matrix = confusion_matrix(y_test, pred_abc)
sb.heatmap(cf_matrix/np.sum(cf_matrix), annot=True, fmt='.2%')
plt.title('Confusion Matrix of AdaBoost Classifier')
plt.xlabel('Predicted')
plt.ylabel('Actual')
```

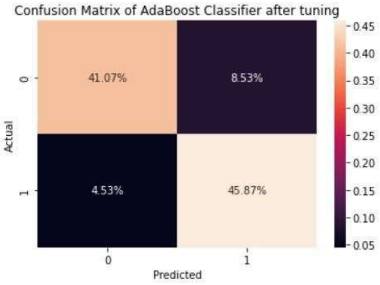
```
Text(33.0, 0.5, 'Actual')
```

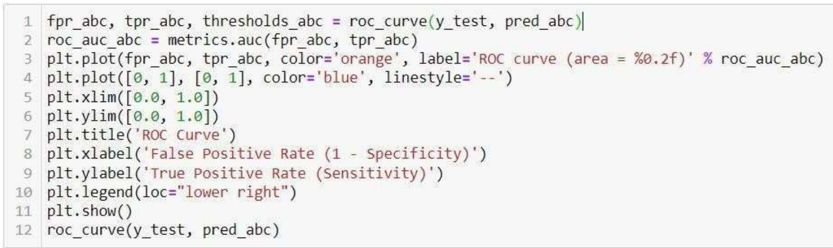




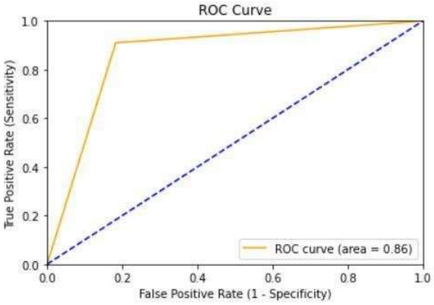
```
cf_matrix = confusion_matrix(y_test, pred_abc_tuned)
sb.heatmap(cf_matrix/np.sum(cf_matrix), annot=True, fmt='.2%')
plt.title('Confusion Matrix of AdaBoost Classifier after tuning')
plt.xlabel('Predicted')
plt.ylabel('Actual')
```

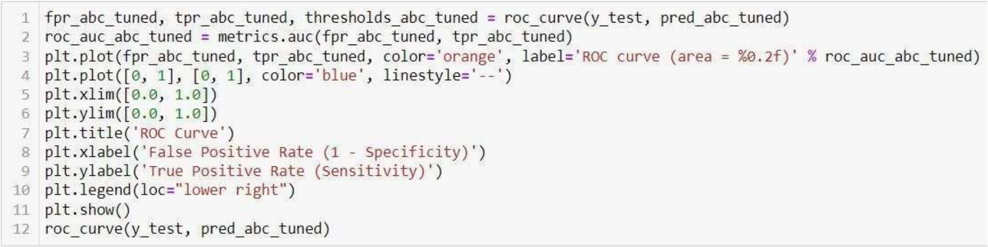
```
Text(33.0, 0.5, 'Actual')
```



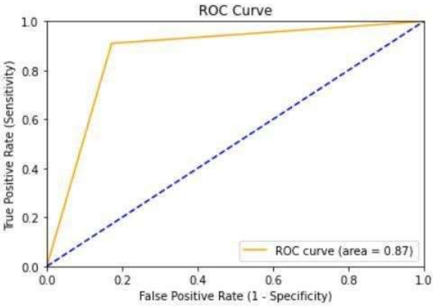


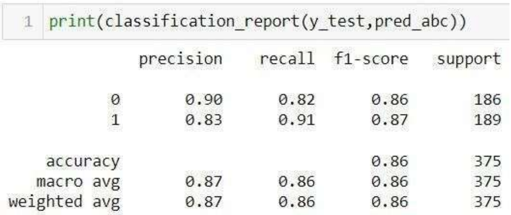
```
fpr_abc, tpr_abc, thresholds_abc = roc_curve(y_test, pred_abc)
roc_auc_abc = metrics.auc(fpr_abc, tpr_abc)
plt.plot(fpr_abc, tpr_abc, color='orange', label='ROC curve (area = %0.2f)' % roc_auc_abc)
plt.plot([0, 1], [0, 1], color='blue', linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.title('ROC Curve')
plt.xlabel('False Positive Rate (1 - Specificity)')
plt.ylabel('True Positive Rate (Sensitivity)')
plt.legend(loc="lower right")
plt.show()
roc_curve(y_test, pred_abc)
```



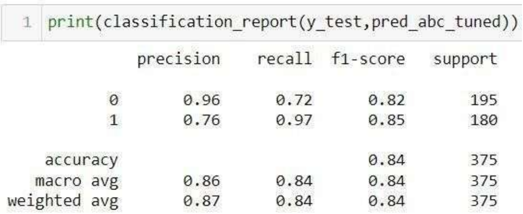


```
fpr_abc_tuned, tpr_abc_tuned, thresholds_abc_tuned = roc_curve(y_test, pred_abc_tuned)
roc_auc_abc_tuned = metrics.auc(fpr_abc_tuned, tpr_abc_tuned)
plt.plot(fpr_abc_tuned, tpr_abc_tuned, color='orange', label='ROC curve (area = %0.2f)' % roc_auc_abc_tuned)
plt.plot([0, 1], [0, 1], color='blue', linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.title('ROC Curve')
plt.xlabel('False Positive Rate (1 - Specificity)')
plt.ylabel('True Positive Rate (Sensitivity)')
plt.legend(loc="lower right")
plt.show()
roc_curve(y_test, pred_abc_tuned)
```

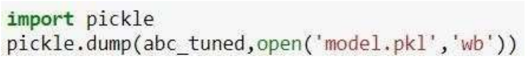




```
print(classification_report(y_test,pred_abc))
              precision    recall  f1-score   support

           0       0.90      0.82      0.86       186
           1       0.83      0.91      0.87       189

    accuracy                           0.86       375
   macro avg       0.87      0.86      0.86       375
weighted avg       0.87      0.86      0.86       375
```

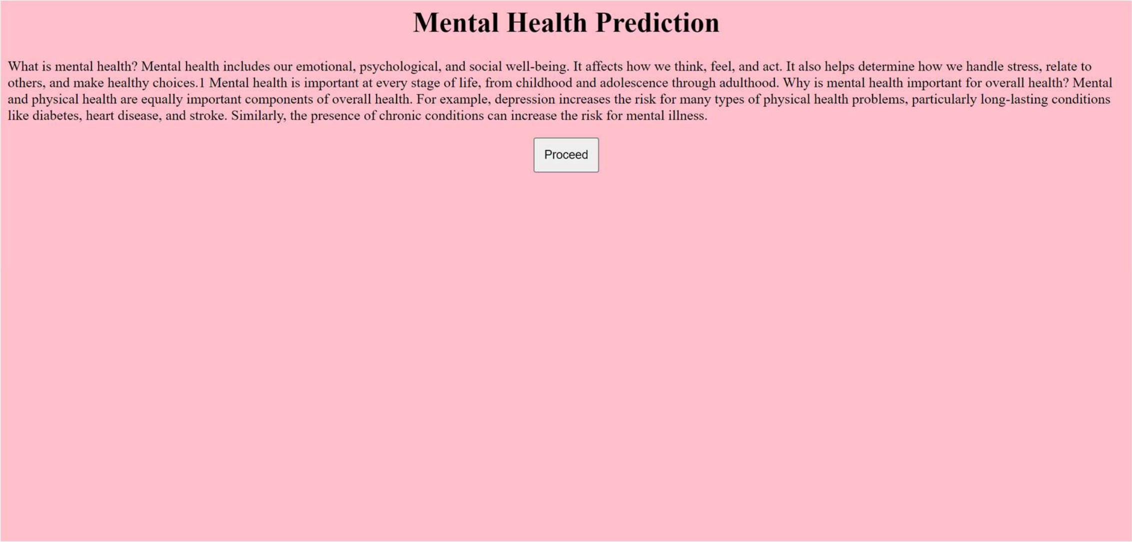


```
print(classification_report(y_test,pred_abc_tuned))
              precision    recall  f1-score   support

           0       0.96      0.72      0.82       195
           1       0.76      0.97      0.85       180

    accuracy                           0.84       375
   macro avg       0.86      0.84      0.84       375
weighted avg       0.87      0.84      0.84       375
```



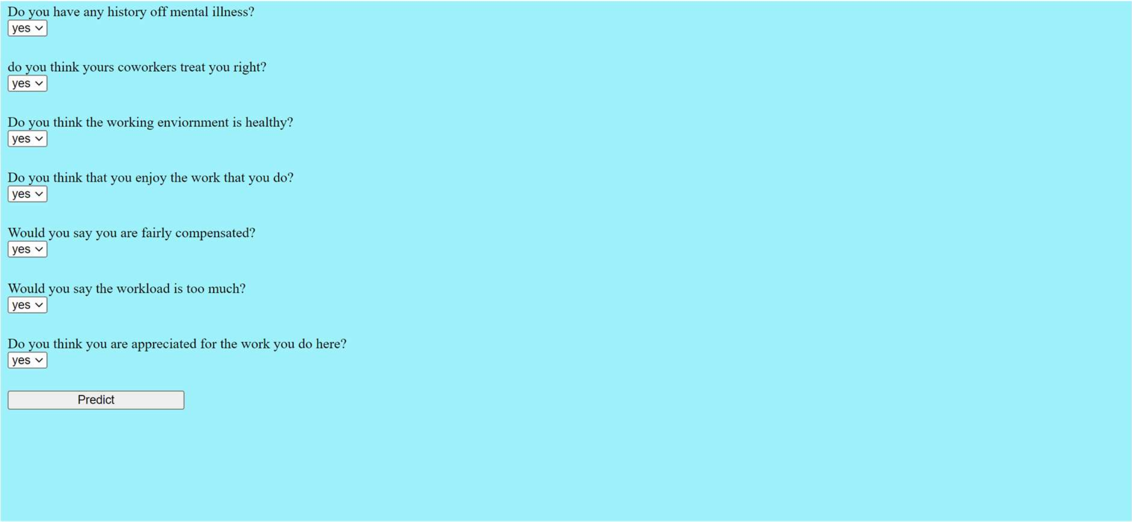
```
import pickle
pickle.dump(abc_tuned,open('model.pkl','wb'))
```

Result



Mental Health Prediction

What is mental health? Mental health includes our emotional, psychological, and social well-being. It affects how we think, feel, and act. It also helps determine how we handle stress, relate to others, and make healthy choices.1 Mental health is important at every stage of life, from childhood and adolescence through adulthood. Why is mental health important for overall health? Mental and physical health are equally important components of overall health. For example, depression increases the risk for many types of physical health problems, particularly long-lasting conditions like diabetes, heart disease, and stroke. Similarly, the presence of chronic conditions can increase the risk for mental illness.



Do you have any history off mental illness?
yes

do you think yours coworkers treat you right?
yes

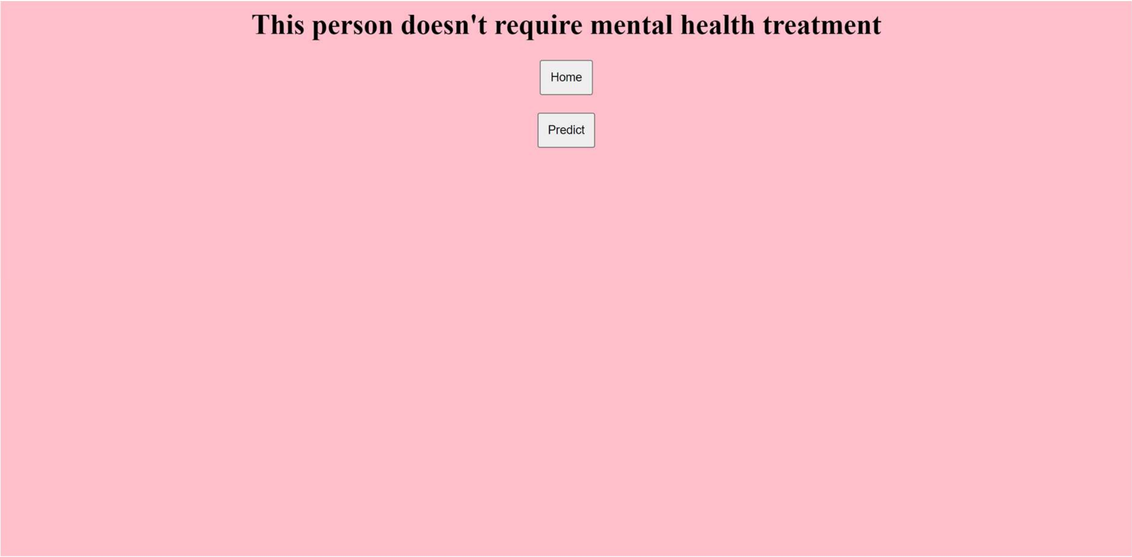
Do you think the working enviornment is healthy?
yes

Do you think that you enjoy the work that you do?
yes

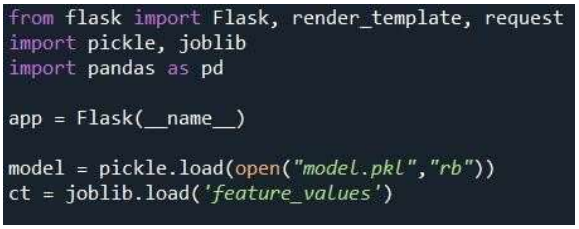
Would you say you are fairly compensated?
yes

Would you say the workload is too much?
yes

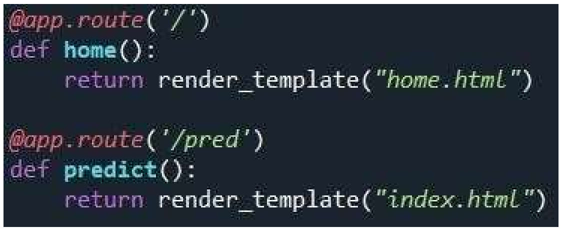
Do you think you are appreciated for the work you do here?
yes



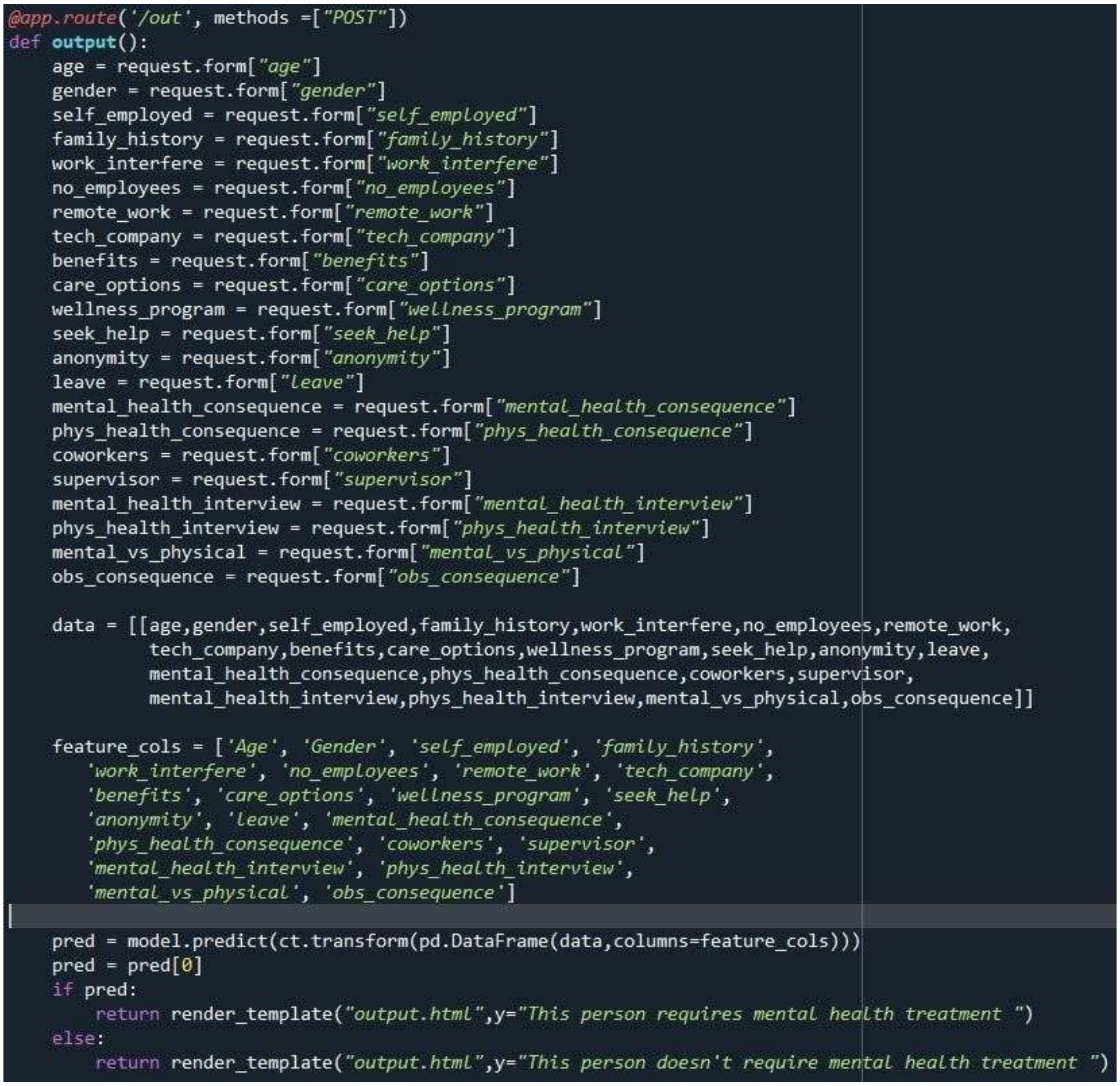
This person doesn't require mental health treatment



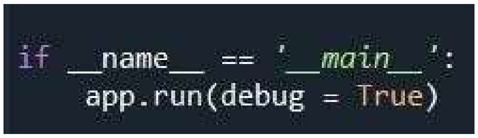
```
from flask import Flask, render_template, request
import pickle, joblib
import pandas as pd

app = Flask(__name__)

model = pickle.load(open("model.pkl","rb"))
ct = joblib.load('feature_values')
```



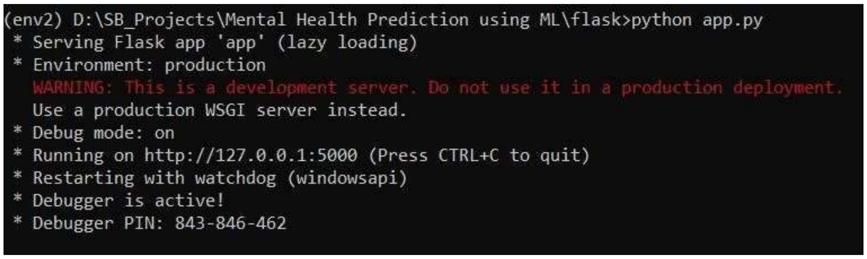
```
@app.route('/')
def home():
    return render_template("home.html")

@app.route('/pred')
def predict():
    return render_template("index.html")
```



```python
@app.route('/out', methods =["POST"])
def output():
    age = request.form["age"]
    gender = request.form["gender"]
    self_employed = request.form["self_employed"]
    family_history = request.form["family_history"]
    work_interfere = request.form["work_interfere"]
    no_employees = request.form["no_employees"]
    remote_work = request.form["remote_work"]
    tech_company = request.form["tech_company"]
    benefits = request.form["benefits"]
    care_options = request.form["care_options"]
    wellness_program = request.form["wellness_program"]
    seek_help = request.form["seek_help"]
    anonymity = request.form["anonymity"]
    leave = request.form["leave"]
    mental_health_consequence = request.form["mental_health_consequence"]
    phys_health_consequence = request.form["phys_health_consequence"]
    coworkers = request.form["coworkers"]
    supervisor = request.form["supervisor"]
    mental_health_interview = request.form["mental_health_interview"]
    phys_health_interview = request.form["phys_health_interview"]
    mental_vs_physical = request.form["mental_vs_physical"]
    obs_consequence = request.form["obs_consequence"]

    data = [[age,gender,self_employed,family_history,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]]

    feature_cols = ['Age', 'Gender', 'self_employed', 'family_history',
       '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']

    pred = model.predict(ct.transform(pd.DataFrame(data,columns=feature_cols)))
    pred = pred[0]
    if pred:
        return render_template("output.html",y="This person requires mental health treatment ")
    else:
        return render_template("output.html",y="This person doesn't require mental health treatment ")
```



```python
if __name__ == '__main__':
    app.run(debug = True)
```



```
(env2) D:\SB_Projects\Mental Health Prediction using ML\flask>python app.py
 * Serving Flask app 'app' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000 (Press CTRL+C to quit)
 * Restarting with watchdog (windowsapi)
 * Debugger is active!
 * Debugger PIN: 843-846-462
```

Advantages and diasadvantages:

In classifying the depression and anxiety cases with machine learning models, the research shows a better result in terms of accuracy for the studies conducted. Most of the research articles show that machine learning models have obtained the accuracy of above 70%. However, Chekroud et al.

- CONS
- There are differences in clinical assessment, and one professional may provide a diagnosis that another disagrees with. This can get confusing or lead to someone having multiple diagnoses.
- There can be a stigma associated with mental health diagnoses.
- Parents or young people may over-identify with an illness model. This may reinforce the problem, e.g., parents do not encourage an anxious child to try new things.
- A diagnosis emphasizes the problem being in the child/person and is less likely to consider other factors, e.g., family and peer interactions.
- Mental health diagnoses are a Western explanation and discount other cultural explanations.

Conclusion

- Many different techniques and algorithms had been introduced and proposed to test and solve the mental health problems. There are still many solutions that can be refined. In addition, there are still many problems to be discovered and tested using a wide variety of settings in machine learning for the mental health domain. As classifying the mental health data is generally a very challenging problem, the features used in the machine learning algorithms will significantly affect the performance of the classification.

- The existing studies and research show that machine learning can be a useful tool in helping understand psychiatric disorders. Besides that, it may also help distinguish and classify the mental health problems among patients for further treatment. Newer approaches that use data that arise from the integration of various sensor modalities present in technologically advanced devices have proven to be a convenient resource to recognize the mood state and responses from patients among others.

- It is noticeable that most of the research and studies are still struggling to validate the results because of insufficiency of acceptable validated evidence, especially from the external sources. Besides that, most of the machine learning might not have the same performance across all the problems. The performance of the machine learning models will vary depending on the data samples obtained and the features of the data. Moreover, machine learning models can also be affected by preprocessing activities such as data cleaning and parameter tuning in order to achieve optimal results.

- Hence, it is very important for researchers to investigate and analyze the data with various machine learning algorithms to choose the highest accuracy among the machine learning algorithms . Not only that, challenges and limitations faced by the researchers need to be managed with proper care to achieve satisfactory results that could improve the clinical practice and decision-making.

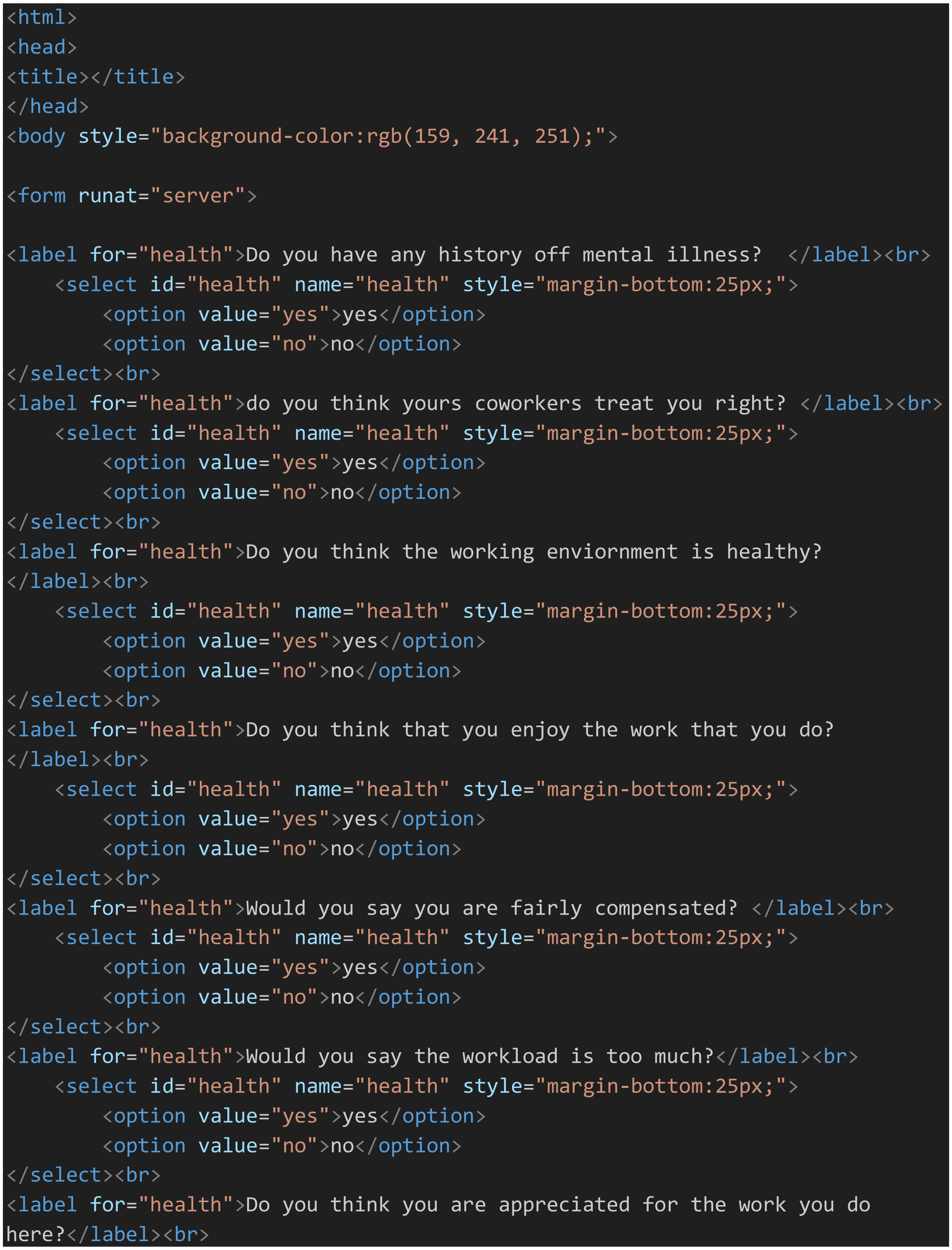
Future scope:

We believe we were able to achieve a good accuracy for each of the four diseases. furthermore, in future we can add more disease and combine multiple method along with questionnaire to make this process more robust and stronger.

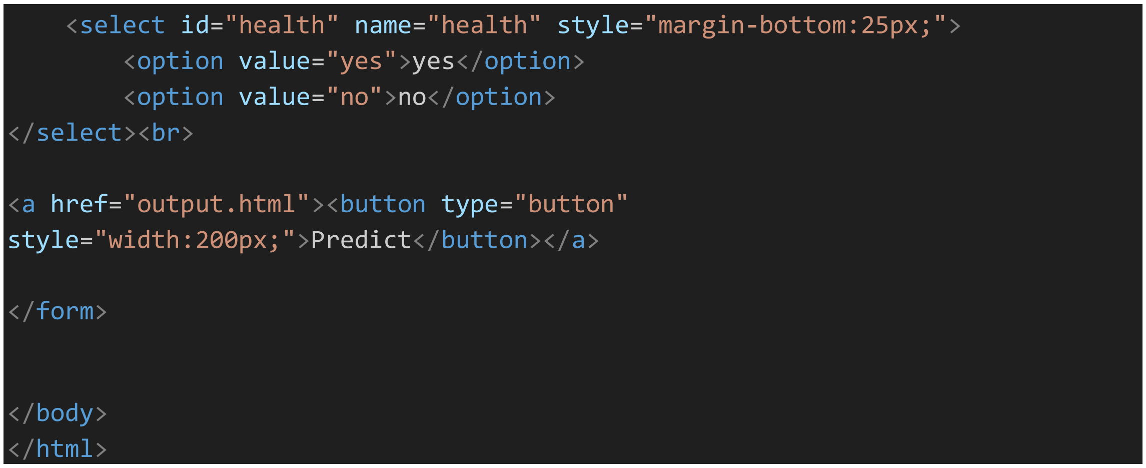
Appendix:

```
<html>

<head>

<title></title>

</head>

<form runat="server">

<body style="background-color:pink;">

<h1 style="text-align:center;">Mental Health Prediction</h1>

<p>What is mental health? Mental health includes our emotional, psychological, and social well-being. It affects how we think, feel, and act. It also helps determine how we handle stress, relate to others, and make healthy choices.1 Mental health is important at every stage of life, from childhood and adolescence through adulthood.

Why is mental health important for overall health? Mental and physical health are equally important components of overall health. For example, depression increases the risk for many types of physical health problems, particularly long-lasting conditions like diabetes, heart disease, and stroke. Similarly, the presence of chronic conditions can increase the risk for mental illness.</p>

<center>

<a href="index.html"><button type="button"
style="padding:10px;">Proceed</button></a>

</center>
```

```
</body>

</html>
```



```
<html>
<head>
<title></title>
</head>
<body style="background-color:rgb(159, 241, 251);">

<form runat="server">

<label for="health">Do you have any history off mental illness?  </label><br>
    <select id="health" name="health" style="margin-bottom:25px;">
        <option value="yes">yes</option>
        <option value="no">no</option>
</select><br>
<label for="health">do you think yours coworkers treat you right? </label><br>
    <select id="health" name="health" style="margin-bottom:25px;">
        <option value="yes">yes</option>
        <option value="no">no</option>
</select><br>
<label for="health">Do you think the working enviornment is healthy?
</label><br>
    <select id="health" name="health" style="margin-bottom:25px;">
        <option value="yes">yes</option>
        <option value="no">no</option>
</select><br>
<label for="health">Do you think that you enjoy the work that you do?
</label><br>
    <select id="health" name="health" style="margin-bottom:25px;">
        <option value="yes">yes</option>
        <option value="no">no</option>
</select><br>
<label for="health">Would you say you are fairly compensated? </label><br>
    <select id="health" name="health" style="margin-bottom:25px;">
        <option value="yes">yes</option>
        <option value="no">no</option>
</select><br>
<label for="health">Would you say the workload is too much?</label><br>
    <select id="health" name="health" style="margin-bottom:25px;">
        <option value="yes">yes</option>
        <option value="no">no</option>
</select><br>
<label for="health">Do you think you are appreciated for the work you do
here?</label><br>
```



```
    <select id="health" name="health" style="margin-bottom:25px;">
        <option value="yes">yes</option>
        <option value="no">no</option>
</select><br>

<a href="output.html"><button type="button"
style="width:200px;">Predict</button></a>

</form>


</body>
</html>
```

```
<html>

<head>

<title></title>

</head>

<body style="background-color:pink;">

<body>

<h1 style="text-align:center;"> This person doesn't require mental health treatment</h1>

<center>

<a href="home.html" style="text-align:center;"><button type="button"
style="padding:10px;">Home</button></a>

</center>

<center>

<a href="home.html"><button type="button" style="margin-top: 20px;
padding:10px;">Predict</button></a>

</center>
```

https://github.com/smartinternz02/SI-GuidedProject-604001-1698062811.git