Sonal Shukla (CS23176)

Lab-5 Assignment-1 : To implement the Random Forest algorithm on the Gaming and Mental Health dataset and evaluate the model performance using appropriate metrics.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,classification_report
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
path="/content/drive/MyDrive/ML_DATASETS/Gaming and Mental Health - Gaming a
df=pd.read_csv(path)
df.head(5)
```

	record_id	age	gender	daily_gaming_hours	game_genre	primary_game	gami
0	GD0001	17	Male	11.1	Mobile Games	Clash of Clans	
1	GD0002	21	Male	3.0	MOBA	Dota 2	
2	GD0003	23	Male	7.6	FPS	CS:GO	
3	GD0004	20	Female	7.2	RPG	Skyrim	
4	GD0005	18	Male	6.8	Battle Royale	Apex Legends	

5 rows × 27 columns

```
df.shape
```

```
(1000, 27)
```

```
df.columns
```

```
Index(['record_id', 'age', 'gender', 'daily_gaming_hours', 'game_genre',
       'primary_game', 'gaming_platform', 'sleep_hours', 'sleep_quality',
       'sleep_disruption_frequency', 'academic_work_performance',
'grades_gpa',
       'work_productivity_score', 'mood_state', 'mood_swing_frequency',
       'withdrawal_symptoms', 'loss_of_other_interests',
       'continued_despite_problems', 'eye_strain', 'back_neck_pain',
       'weight_change_kg', 'exercise_hours_weekly',
'social_isolation_score',
```

```
       'face_to_face_social_hours_weekly', 'monthly_game_spending_usd',
       'years_gaming', 'gaming_addiction_risk_level'],
      dtype='object')
```

```
df.isnull().any().any()
```

```
np.True_
```

```
df[df.isnull().any(axis = 1)]
```

	record_id	age	gender	daily_gaming_hours	game_genre	primary_game	ga
0	GD0001	17	Male	11.1	Mobile Games	Clash of Clans	
2	GD0003	23	Male	7.6	FPS	CS:GO	
4	GD0005	18	Male	6.8	Battle Royale	Apex Legends	
7	GD0008	28	Female	10.3	Mobile Games	Clash of Clans	
8	GD0009	18	Male	1.4	Mobile Games	Genshin Impact	
...	...	...	...	...	...	...	
995	GD0996	15	Female	8.6	Strategy	Age of Empires	
996	GD0997	18	Male	5.4	MMO	Elder Scrolls Online	
997	GD0998	23	Male	7.3	RPG	Skyrim	
998	GD0999	18	Male	3.1	Strategy	StarCraft II	
999	GD1000	29	Male	3.5	RPG	Cyberpunk 2077	

572 rows × 27 columns

```
X = df.drop(columns=['gaming_addiction_risk_level'])
y = df['gaming_addiction_risk_level']
```

```
X.head()
```

	record_id	age	gender	daily_gaming_hours	game_genre	primary_game	gami
0	GD0001	17	Male	11.1	Mobile Games	Clash of Clans	
1	GD0002	21	Male	3.0	MOBA	Dota 2	
2	GD0003	23	Male	7.6	FPS	CS:GO	
3	GD0004	20	Female	7.2	RPG	Skyrim	
4	GD0005	18	Male	6.8	Battle Royale	Apex Legends	

5 rows × 26 columns

```
X = pd.get_dummies(X,dtype = int)
```

```
X.head()
```

	age	daily_gaming_hours	sleep_hours	grades_gpa	work_productivity_score
0	17	11.1	3.7	1.25	NaN
1	21	3.0	7.2	3.75	2.0
2	23	7.6	4.4	NaN	9.0
3	20	7.2	5.1	1.62	2.0
4	18	6.8	3.4	2.44	NaN

5 rows × 1084 columns

```
X.shape
```

```
(1000, 1084)
```

```
X.columns
```

```
Index(['age', 'daily_gaming_hours', 'sleep_hours', 'grades_gpa',
       'work_productivity_score', 'withdrawal_symptoms',
       'loss_of_other_interests', 'continued_despite_problems',
'eye_strain',
       'back_neck_pain',
       ...
       'mood_state_Excited', 'mood_state_Irritable', 'mood_state_Normal',
       'mood_state_Restless', 'mood_state_Withdrawn',
       'mood_swing_frequency_Daily', 'mood_swing_frequency_Never',
       'mood_swing_frequency_Often', 'mood_swing_frequency_Rarely',
       'mood_swing_frequency_Sometimes'],
      dtype='object', length=1084)
```

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,train_size=0.7,test_siz
X_train.shape,X_test.shape
```

```
((700, 1084), (300, 1084))
```

```
from sklearn.ensemble import RandomForestClassifier
rf_classifier = RandomForestClassifier(n_estimators=100,n_jobs=-1,max_depth=
rf_classifier.fit(X_train,y_train)
```

```
RandomForestClassifier
RandomForestClassifier(max_depth=5, n_jobs=-1, oob_score=True, random_state=4
```

```
y_pred = rf_classifier.predict(X_test)
```

```
from sklearn.metrics import confusion_matrix,accuracy_score,classification_r
```

```
print('Confusion Matrix')
cm=confusion_matrix(y_test,y_pred)
print(confusion_matrix(y_test,y_pred))
```

```
Confusion Matrix
[[ 21   9  17   5]
 [  0 148   0   0]
 [  8  38   4   0]
 [  8   1   7  34]]
```

```
print('Accuracy Score')
print(accuracy_score(y_test,y_pred))
```

```
Accuracy Score
0.69
```

```
rf_classifier.oob_score_
```

```
0.6914285714285714
```

```
print('Classification Report')
print(classification_report(y_test,y_pred))
```

```
Classification Report
              precision    recall  f1-score   support

        High       0.57      0.40      0.47        52
         Low       0.76      1.00      0.86       148
    Moderate       0.14      0.08      0.10        50
      Severe       0.87      0.68      0.76        50

    accuracy                           0.69       300
```

```
   macro avg       0.58      0.54      0.55       300
weighted avg       0.64      0.69      0.65       300
```

```
rf=RandomForestClassifier(random_state=42,n_jobs=-1)
params={
    'max_depth':[2,3,5],
    'min_samples_leaf':[5,10,20],
    'n_estimators':[10,25,30,50]
}
from sklearn.model_selection import GridSearchCV
grid_search = GridSearchCV(estimator=rf, param_grid=params, cv = 4, n_jobs=-
grid_search.fit(X_train,y_train)
```

```
Fitting 4 folds for each of 36 candidates, totalling 144 fits
```

▸ **GridSearchCV**

▸ **best_estimator_: RandomForestClassifier**

▸ RandomForestClassifier

```
grid_search.best_score_
```

```
np.float64(0.6185714285714285)
```

```
print(grid_search.best_params_)
```

```
{'max_depth': 5, 'min_samples_leaf': 5, 'n_estimators': 50}
```