```
import pandas as pd
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from lightgbm import LGBMClassifier
```

```
df = pd.read_csv("data.csv")


df.drop(columns=["SessionID", "SessionDate"], inplace=True)
```

```
label_encoders = {}
for col in ["StudentLevel", "Discipline", "TaskType", "FinalOutcome"]:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    label_encoders[col] = le
```

```


df["UsedAgain"] = df["UsedAgain"].astype(int)


X = df.drop(columns=["FinalOutcome"])
y = df["FinalOutcome"]
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
param_grid = {
    'num_leaves': [31, 50],
    'max_depth': [-1, 10, 20],
    'learning_rate': [0.05, 0.1],
    'n_estimators': [100, 200]
}
```

```
lgb_model = LGBMClassifier(random_state=42)
grid_search = GridSearchCV(estimator=lgb_model, param_grid=param_grid, cv=5, n_jobs=-1, verbose=1)
grid_search.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 24 candidates, totalling 120 fits
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead of testing was 0.000236 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 355
[LightGBM] [Info] Number of data points in the train set: 8000, number of used features: 8
[LightGBM] [Info] Start training from score -0.734490
[LightGBM] [Info] Start training from score -1.845160
[LightGBM] [Info] Start training from score -2.619010
[LightGBM] [Info] Start training from score -1.240032
```

```
▸ GridSearchCV
▸ best_estimator_: LGBMClassifier
  ▸ LGBMClassifier
```

```
best_model = grid_search.best_estimator_
y_pred = best_model.predict(X_test)
```

```
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, target_names=label_encoders['FinalOutcome'].classes_)
```

```
print(" Best Hyperparameters:", grid_search.best_params_)
print("\n Accuracy:", accuracy)
print("\n Confusion Matrix:\n", conf_matrix)
print("\n Classification Report:\n", class_report)
```

```
Best Hyperparameters: {'learning_rate': 0.05, 'max_depth': -1, 'n_estimators': 100, 'num_leaves': 31}

Accuracy: 0.479

Confusion Matrix:
[[752  78   4  96]
[184 105   8  52]
[ 94  52   4  20]
[409  43   2  97]]

Classification Report:
                      precision    recall  f1-score   support

Assignment Completed       0.52      0.81      0.63       930
            Confused       0.38      0.30      0.33       349
             Gave Up       0.22      0.02      0.04       170
        Idea Drafted       0.37      0.18      0.24       551

            accuracy                           0.48      2000
           macro avg       0.37      0.33      0.31      2000
        weighted avg       0.43      0.48      0.42      2000
```

```
Fitting 5 folds for each of 24 candidates, totalling 120 fits
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead of testing was 0.000228 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 355
[LightGBM] [Info] Number of data points in the train set: 8000, number of used features: 8
[LightGBM] [Info] Start training from score -0.734490
[LightGBM] [Info] Start training from score -1.845160
[LightGBM] [Info] Start training from score -2.619010
[LightGBM] [Info] Start training from score -1.240032
🎯 Best Hyperparameters: {'learning_rate': 0.05, 'max_depth': -1, 'n_estimators': 100, 'num_leaves': 31}

✅ Accuracy: 0.479

🧮 Confusion Matrix:
 [[752  78   4  96]
 [184 105   8  52]
 [ 94  52   4  20]
 [409  43   2  97]]

📊 Classification Report:
                      precision    recall  f1-score   support

Assignment Completed       0.52      0.81      0.63       930
            Confused       0.38      0.30      0.33       349
             Gave Up       0.22      0.02      0.04       170
        Idea Drafted       0.37      0.18      0.24       551

            accuracy                           0.48      2000
           macro avg       0.37      0.33      0.31      2000
        weighted avg       0.43      0.48      0.42      2000
```