รายวิชา ปัญญาประดิษฐ์ (01418261) โดยนางสาว สรัลรัตน์ สนสุรัตน์ 6610402248

1. ทำการโหลดข้อมูลจากชุดข้อมูล psycology แล้วนำมาแสดงเป็นตารางโดยที่ตารางนี้มี 502 columns และมี 11 feature

```
patient = pd.read_excel('Psycology.xls')
patient.head()
```
✓ 0.0s Python

	เพศ	อายุ	เคยมีแฟนมาแล้ว (คน)	จำนวนครั้งที่ไปออกกำลังกายต่อสัปดาห์	ระยะเวลาการนอน	นิสัยการกินอาหาร	เคยคิดฆ่าตัวตาย	จำนวนชั่วโมงที่เข้าห้องสมุดในหนึ่งสัปดาห์	ระดับความเครียดด้านการเงิน	เคยกินยานอนหลับ	ภาวะซึมเศร้า
0	Male	28	5	3	5-6 ชั่วโมง	อาหารสุขภาพ	Yes	8	3	Yes	Yes
1	Male	23	5	2	มากกว่า 8 ชั่วโมง	อาหารทั่วไป	No	10	4	No	Yes
2	Female	23	1	3	น้อยกว่า 5 ชั่วโมง	อาหารสุขภาพ	Yes	0	3	No	No
3	Female	20	5	5	มากกว่า 8 ชั่วโมง	Junkfood	Yes	2	5	No	Yes
4	Male	29	4	3	มากกว่า 8 ชั่วโมง	Junkfood	Yes	1	3	No	Yes

```
patient.shape
```
✓ 0.0s Python

```
(502, 11)
```

โดยที่ตัวแปรเป้าหมายคือคอลัมน์ "ภาวะซึมเศร้า" โดยที่ข้อมูลอื่นเป็นตัวแปรนำเข้า (Features)

```
patient.columns
```
✓ 0.0s Python

```
Index(['เพศ', 'อายุ', 'เคยมีแฟนมาแล้ว  (คน)',
       'จำนวนครั้งที่ไปออกกำลังกายต่อสัปดาห์', 'ระยะเวลาการนอน',
       'นิสัยการกินอาหาร', 'เคยคิดฆ่าตัวตาย',
       'จำนวนชั่วโมงที่เข้าห้องสมุดในหนึ่งสัปดาห์',
       'ระดับความเครียดด้านการเงิน', 'เคยกินยานอนหลับ', 'ภาวะซึมเศร้า'],
      dtype='object')
```

```
patient['ภาวะซึมเศร้า'].value_counts()
```
✓ 0.0s Python

```
ภาวะซึมเศร้า
Yes    252
No     250
Name: count, dtype: int64
```

ทำการแปลงข้อมูล (Encoding) จากตัวหนังสือเป็นตัวเลขเพื่อนำไปใช้ในการทำนาย

```
col_names = patient.columns.tolist()
feature_cols = col_names[:-1]
X = patient[feature_cols]
y = patient['ภาวะซึมเศร้า']

# แปลงข้อมูลหมวดหมู่ใน X ให้เป็น One-Hot Encoding
X_encoded = pd.get_dummies(X, drop_first=True)

# แปลงข้อมูลเป้าหมาย (Target Variable) ให้เป็นตัวเลข (ถ้าจำเป็น)
labelencoder = LabelEncoder()
y_encoded = labelencoder.fit_transform(y)
```
✓ 0.0s Python

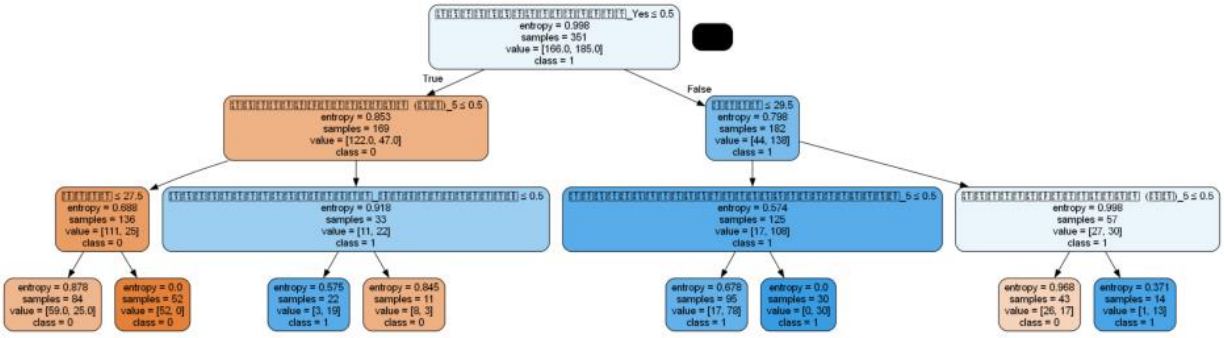
การปรับ Hyperparameter: ใช้ Optuna เพื่อค้นหาค่า max_depth, min_samples_split, และ min_samples_leaf ที่เหมาะสมที่สุดสำหรับ DecisionTreeClassifier และทำการแก้ปัญหาความไม่สมดุลของข้อมูลโดยใช้ SMOTE เพื่อปรับสมดุลข้อมูลนำข้อมูลทั้งหมดมาแบ่งเป็นชุด train และ test และทำการหาค่าประสิทธิภาพ

```
# แบ่งข้อมูลออกเป็นชุดฝึก (train) และชุดทดสอบ (test)
X_train, X_test, y_train, y_test = train_test_split(X_encoded, y_encoded, test_size=0.3, random_state=1)

# สร้างโมเดล Decision Tree และฝึกโมเดล
clf = DecisionTreeClassifier(criterion="entropy", max_depth=3)
clf = clf.fit(X_train, y_train)

# ทดสอบความแม่นยำบนชุดทดสอบ
y_pred = clf.predict(X_test)
print(classification_report(y_test, y_pred))
print("Accuracy:", metrics.accuracy_score(y_test, y_pred))

# ประเมินผลด้วย Cross-Validation
scores = cross_val_score(clf, X_encoded, y_encoded, cv=5)
print("Cross-Validation Accuracy:", scores.mean())
```
✓ 0.0s Python

	precision	recall	f1-score	support
0	0.77	0.80	0.78	84
1	0.73	0.70	0.72	67
accuracy			0.75	151
macro avg	0.75	0.75	0.75	151
weighted avg	0.75	0.75	0.75	151

Accuracy: 0.7549668874172185
Cross-Validation Accuracy: 0.7710693069306931

ค่าประสิทธิภาพและอื่นๆที่ได้จากการคำนวณ



ภาพต้นไม้ที่ได้จากการคำนวณ