

การบ้าน Image Classification

```
▼ นำเข้า library modules ที่ต้องใช้ในการสร้าง models

[82] from sklearn.metrics import accuracy_score, log_loss
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
import matplotlib.pyplot as plt

▼ ประกาศตัวแปร list ไว้เก็บค่า accuracy และ ค่า loss ของ models ทั้งหมด

[83] acc_values = []
loss_values = []
model_names = ["Decision Tree", "KNN"]

▼ เตรียมข้อมูล train และ test

[84] x_train, x_test, y_train, y_test = train_test_split(train_images, train_labels, test_size=0.2)
```

```
+ โฉด + ข้อความ RAM 100% 100%
▼ ประกาศตัวแปร list ไว้เก็บค่า accuracy และ ค่า loss ของ models ทั้งหมด

[83] acc_values = []
loss_values = []
model_names = ["Decision Tree", "KNN"]

▼ เตรียมข้อมูล train และ test

[84] x_train, x_test, y_train, y_test = train_test_split(train_images, train_labels, test_size=0.2)

▼ เปลี่ยน shape ของข้อมูล train และ test ใหม่ให้เป็น array 2 มิติเพื่อนำข้อมูลไป train

[85] # ไม่สามารถ train ข้อมูล array 3 มิติได้ต้องแปลงเป็น array 2 มิติ ก่อน
x_train = x_train.reshape(x_train.shape[0], -1)
x_test = x_test.reshape(x_test.shape[0], -1)
y_train = y_train.reshape(y_train.shape[0], -1)
y_test = y_test.reshape(y_test.shape[0], -1)
```

```
▼ ML Model อันแรกที่ใช้คือ Decision Tree Model

▼ train model สำหรับ model DecisionTree

[86] dc_model = DecisionTreeClassifier()
dc_model.fit(x_train, y_train)
```

DecisionTreeClassifier

▼ ให้ model ทำนายว่าควรอยู่ class ไหน

```
[87] # model ทำนาย
y_pred = dc_model.predict(x_test)

# แสดงผลการทำนาย
print(np.argmax(y_pred[0]))
print(class_names[np.argmax(y_pred[0])])
```

0
T-shirt/top

▼ ให้ model ทำนายค่าความน่าจะเป็นของการทำนาย

```
[88] # model ทำนายค่าความน่าจะเป็น
y_pred_prob = dc_model.predict_proba(x_test)
# แสดงผลค่าที่ได้มา
print(np.argmax(y_pred_prob))
```

4

▼ วัดค่า accuracy และ ค่า loss ของ decision tree

```
[89] # คำนวณค่า accuracy และค่า loss ของตัว model
acc = accuracy_score(y_test, y_pred)
loss = log_loss(y_test, y_pred_prob)

# แสดงผลค่า accuracy และค่า loss
print(f"Accuracy: {acc:.2f}")
print(f"Loss: {loss:.2f}")

# เก็บค่าที่คำนวณมาได้เพิ่มเข้าไปใน list
acc_values.append(acc)
loss_values.append(loss)
```

Accuracy: 0.80
Loss: 7.36

▼ ML Model อันที่สองที่ใช้คือ KNN

▼ train model สำหรับ model KNN

```
[90] knn_model = KNeighborsClassifier(n_neighbors=5, p=2)
knn_model.fit(x_train, y_train)
```

/usr/local/lib/python3.11/dist-packages/sklearn/neighbors/_classification.py:239: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y
return self._fit(X, y)

KNeighborsClassifier

▼ ให้ model ทำนายว่าอยู่ class ไหน

```
[91] # model ทำนาย
y_pred = knn_model.predict(x_test)

# แสดงผลการทำนาย
print(np.argmax(y_pred[0]))
print(class_names[np.argmax(y_pred[0])])
```

0
T-shirt/top

▼ ให้ model ทำนายค่าความน่าจะเป็นของการทำนาย

```
# model ทำนายค่าความน่าจะเป็น
y_pred_prob = knn_model.predict_proba(x_test)
# แสดงผลค่าที่ได้มา
print(np.argmax(y_pred_prob))
```

17

วัดค่า accuracy และ ค่า loss ของ KNN

```
[93] # คำนวณค่า accuracy และค่า loss ของตัว model
acc = accuracy_score(y_test, y_pred)
loss = log_loss(y_test, y_pred_prob)

# แสดงผลค่า accuracy และค่า loss
print(f"Accuracy: {acc:.2f}")
print(f"Loss: {loss:.2f}")

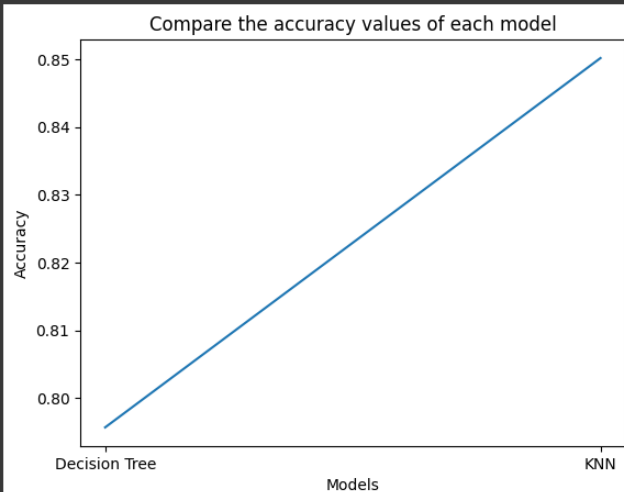
# เก็บค่าที่คำนวณมาได้เพิ่มเข้าไปใน list
acc_values.append(acc)
loss_values.append(loss)
```

Accuracy: 0.85
Loss: 1.75

plot กราฟเส้นเปรียบเทียบวัดค่า accuracy และ ค่า loss ของทั้ง 2 models

กราฟเปรียบเทียบค่า accuracy

```
[94] plt.title("Compare the accuracy values of each model")
plt.plot(model_names, acc_values)
plt.xlabel("Models")
plt.ylabel("Accuracy")
plt.show()
```



กราฟเปรียบเทียบค่า loss

```
[95] plt.title("Compare the loss values of each model")
plt.plot(model_names, loss_values)
plt.xlabel("Models")
plt.ylabel("Loss")
plt.show()
```

