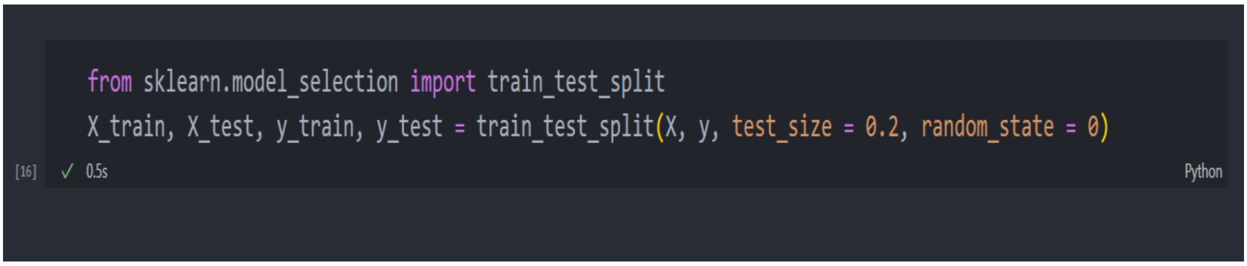
Lab 5.3 Classification (Naive Bayes & Decision Tree) (20/5/2022)
ລະຫັດນັກສຶກສາ: 205Q0010.19
ຊື່ ແລະ ນາມສະກຸນ: ທ. ນູຊື່ວ ເຮີ 3CW1

ຈົ່ງຕອບຄຳຖາມຕໍ່ໄປນີ້ໃຫ້ສຳເລັດດ້ວຍການນຳໃຊ້ຄຳສັ່ງຂອງ Python:

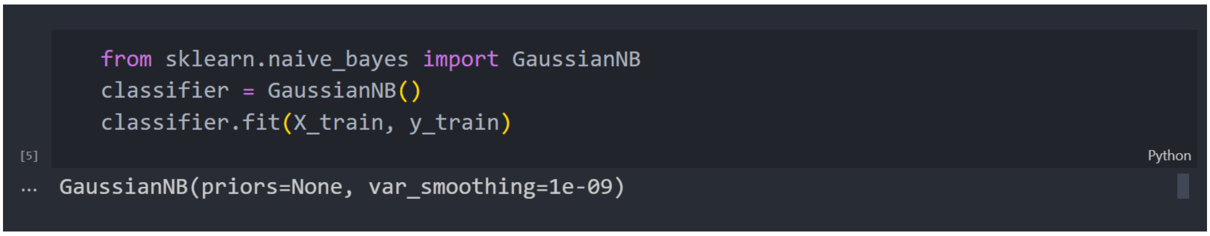
ພາກທີ 1

1.1. ຈົ່ງແຍກຊຸດຂໍ້ມູນSocial_Network_Ads.csvອອກເປັນສອງພາກສ່ວນຄື: ຊຸດຮຽນ 80 ແລະ ຊຸດທົດສອບ 20?



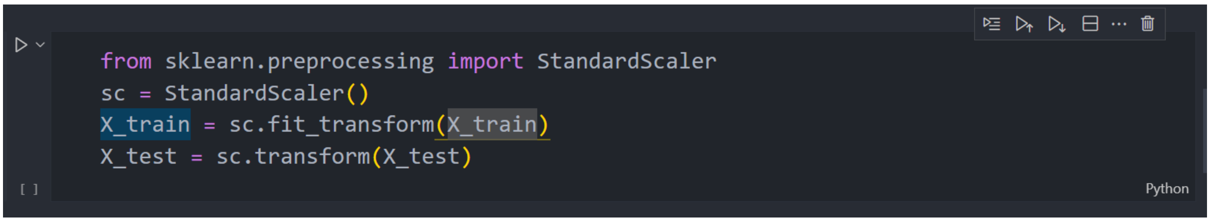
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
```

1.2. ຈົ່ງສ້າງNaive Bayes model ແລະ ທຳການປະມອນຜົນ (fit) ຊຸດຂໍ້ມູນ (X_train, y_train)



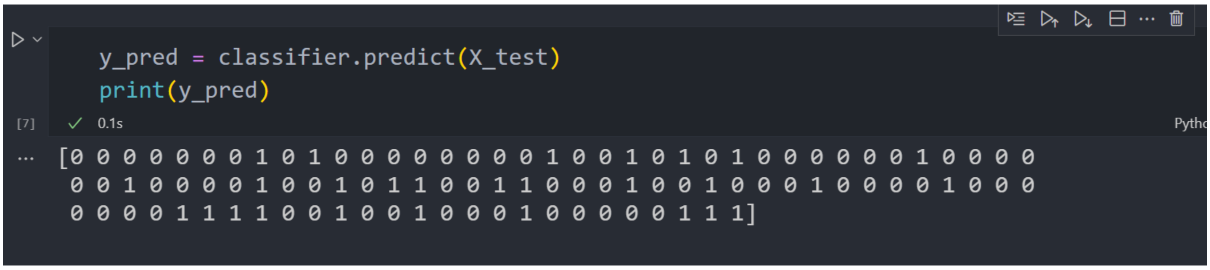
```
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
classifier.fit(X_train, y_train)
```

```
GaussianNB(priors=None, var_smoothing=1e-09)
```



```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

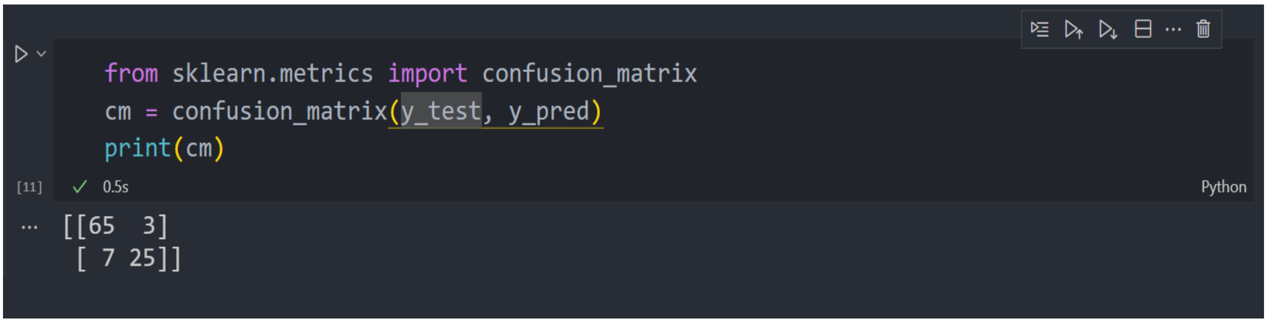
1.3. ຈົ່ງທົດສອບໂມເດວດ້ວຍການpredict(X_test).



```
y_pred = classifier.predict(X_test)
print(y_pred)
```

```
[0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 1 0 0 0 0 0 0 1 0 0 0 0
 0 0 1 0 0 0 0 1 0 0 1 0 1 1 0 0 1 1 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0
 0 0 0 0 1 1 1 1 0 0 1 0 0 1 0 0 0 1 0 0 0 0 0 1 1 1]
```

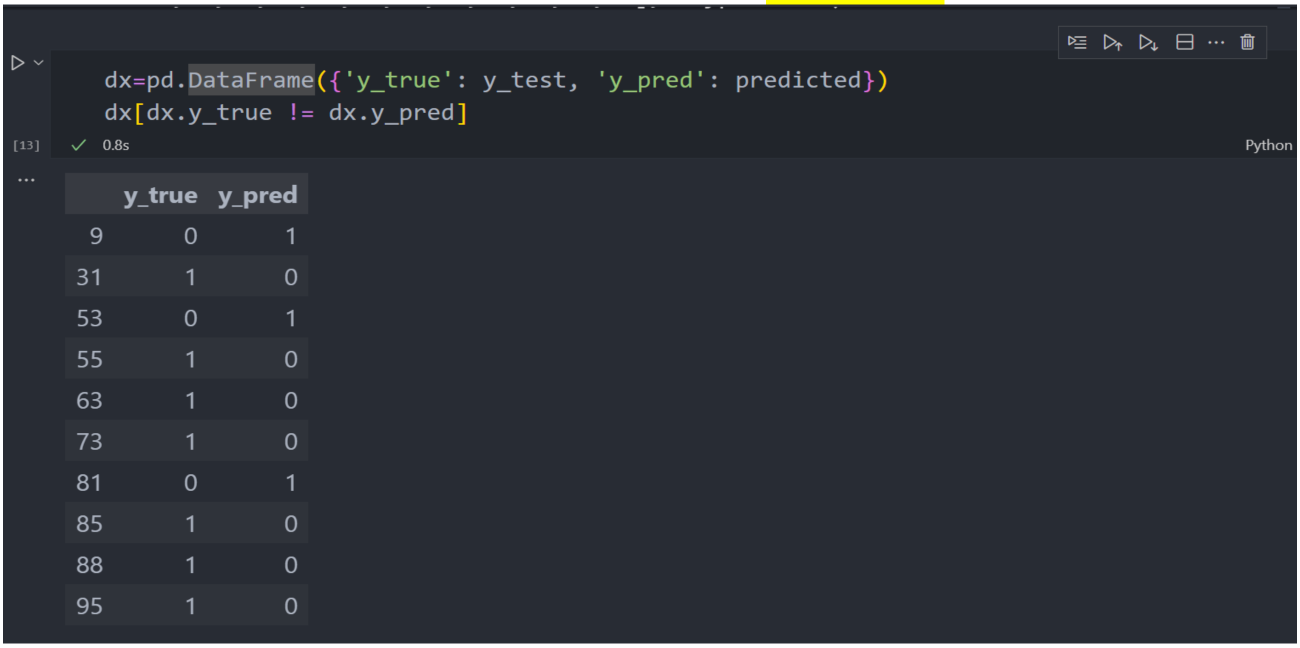
1.4. ຈົ່ງທຳການprocessing ດ້ວຍconfusion_matrix, ກຳນົດTP, TN, FP, FN



```python
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
[[65  3]
 [ 7 25]]
```

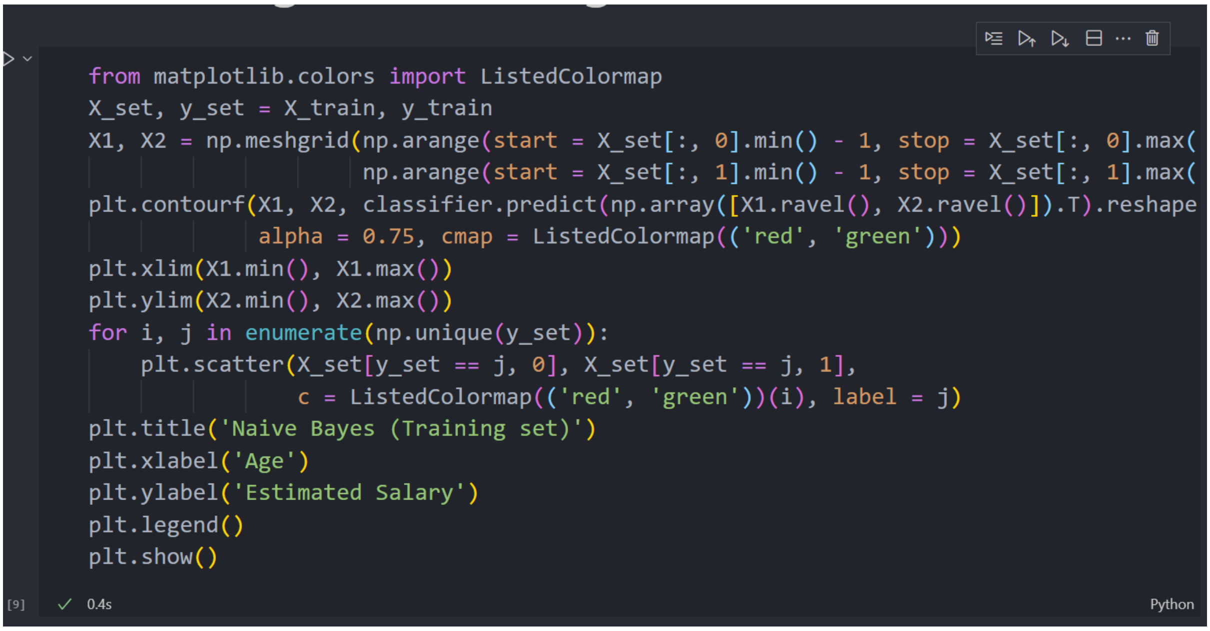
1.5. ຈົ່ງສະແດງຜົນດ້ວຍການສົມທຽບຄ່າຈິງ ແລະ ຄ່າຄາດເດົາຂອງ y_test ດ້ວຍຮູບDataFrame



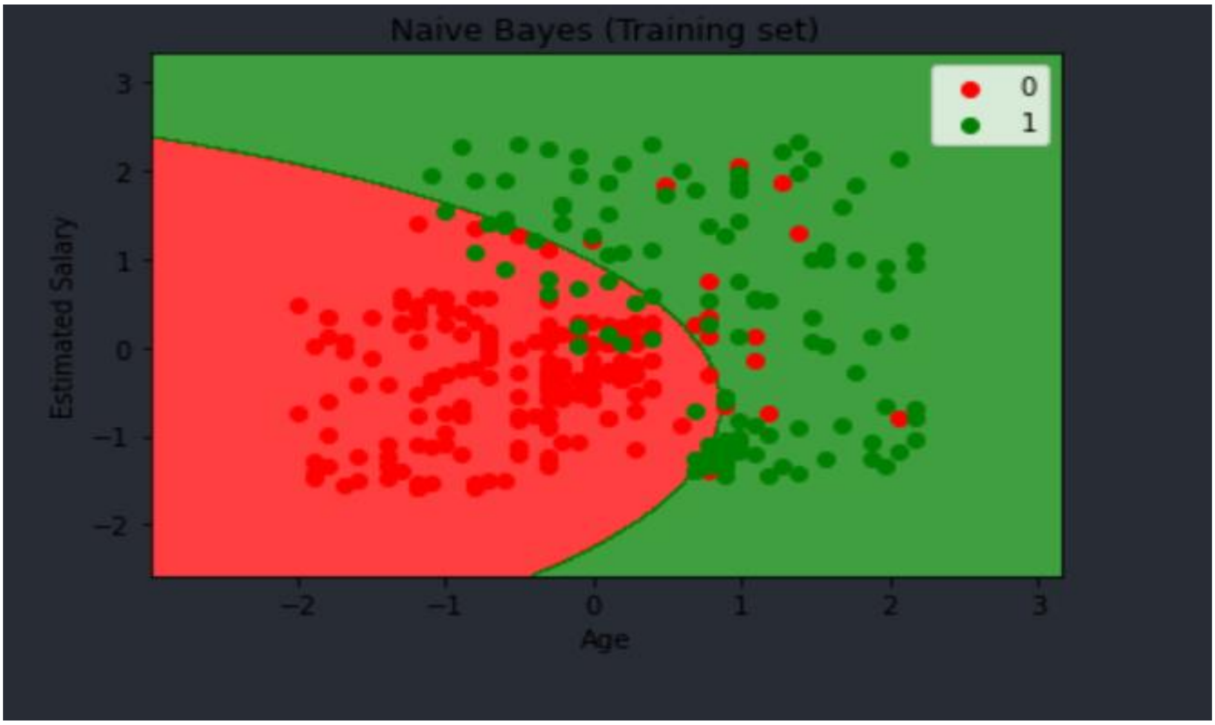
```python
dx=pd.DataFrame({'y_true': y_test, 'y_pred': predicted})
dx[dx.y_true != dx.y_pred]
```

	y_true	y_pred
9	0	1
31	1	0
53	0	1
55	1	0
63	1	0
73	1	0
81	0	1
85	1	0
88	1	0
95	1	0

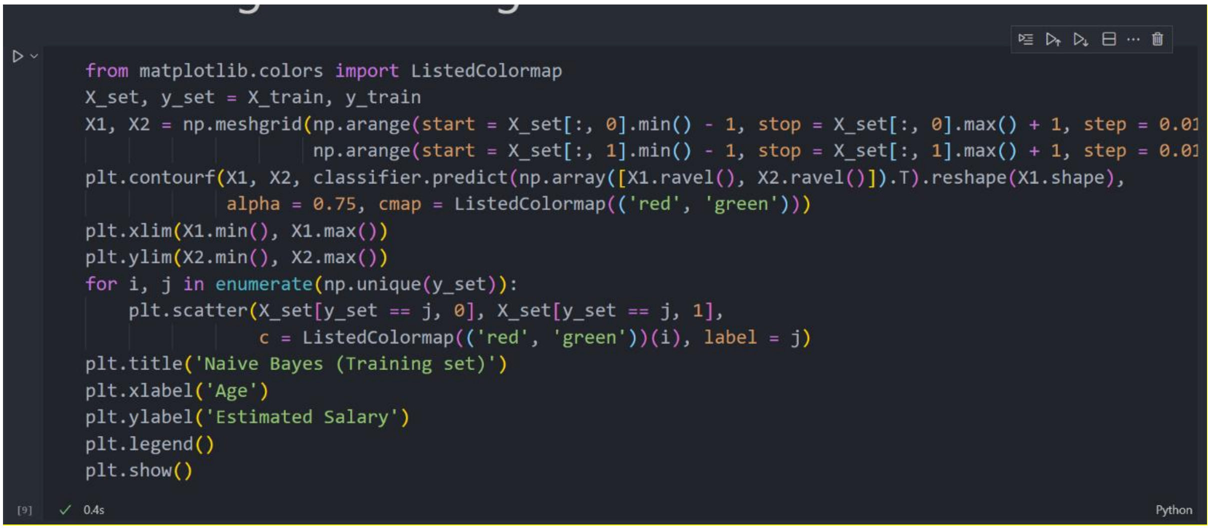
1.6. ຈົ່ງສະແດງຂໍ້ມູນຊຸດຮຽນ (X_train, y_train)ດ້ວຍGraph ບົນພື້ນຖານຊຸດຄຳສັ່ງ matplotlib.



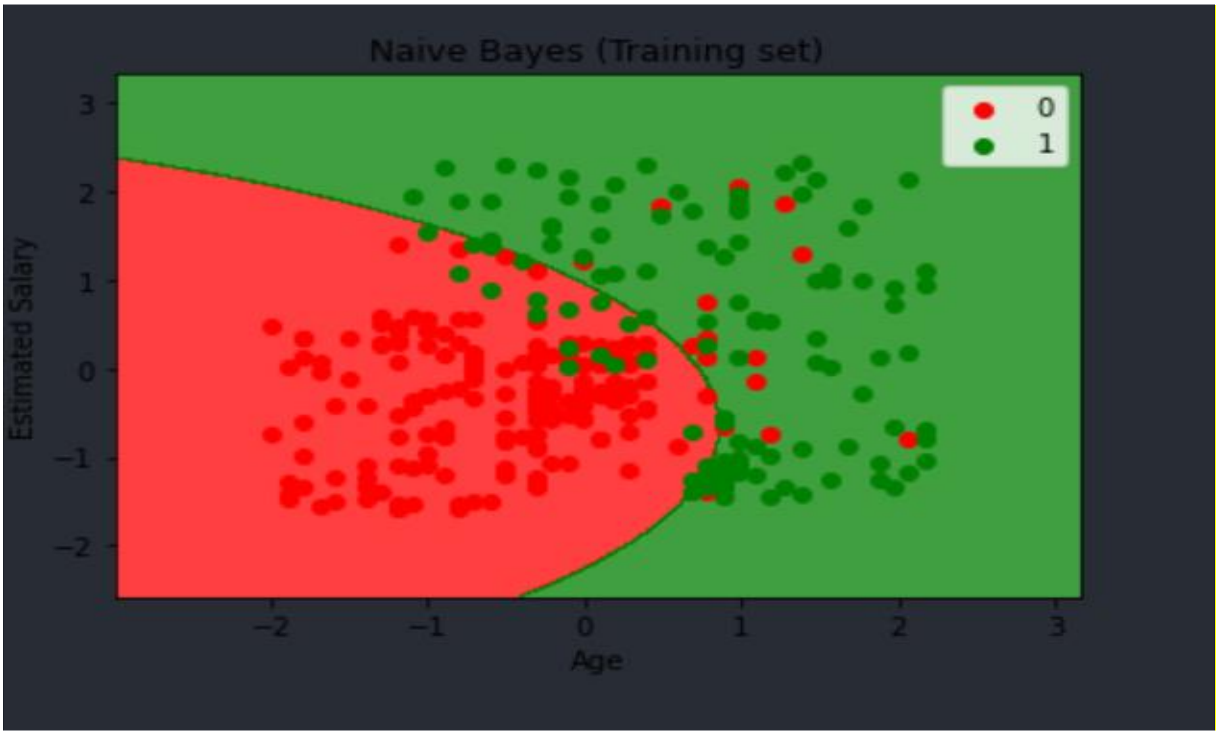
```python
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max(
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max(
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('Naive Bayes (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```



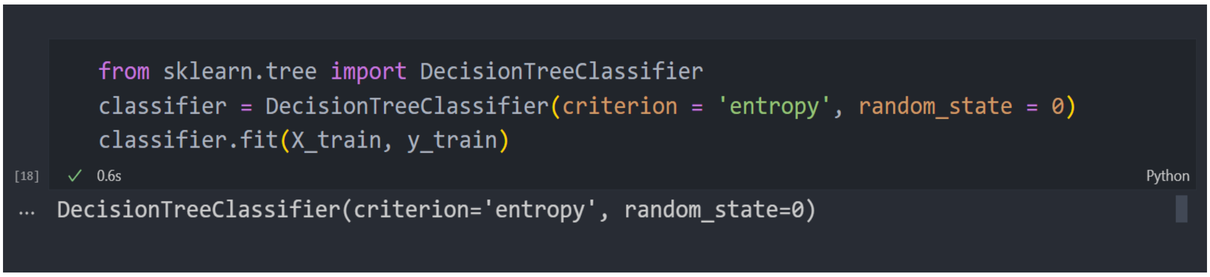
1.7. ຈົ່ງສະແດງຂໍ້ມູນຊຸດຮຽນ (X_test, y_test) ດ້ວຍGraph ບົນພື້ນຖານຊຸດຄຳສັ່ງ matplotlib



```
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('Naive Bayes (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

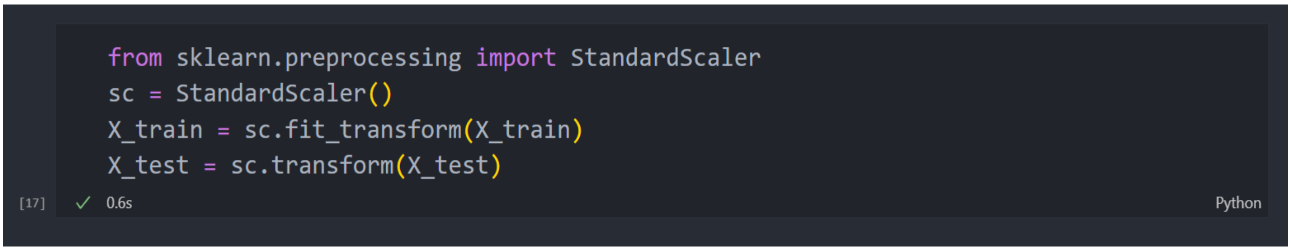


1.8. ຈົ່ງສ້າງDecision Tree Classification model ແລະ ທຳການປະມວນຜົນ (fit) ຊຸດຂໍ້ມູນX_train



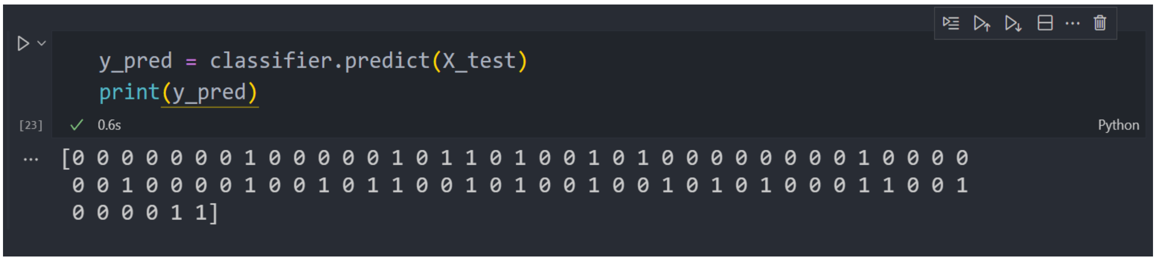
```
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
classifier.fit(X_train, y_train)
```

```
... DecisionTreeClassifier(criterion='entropy', random_state=0)
```



```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

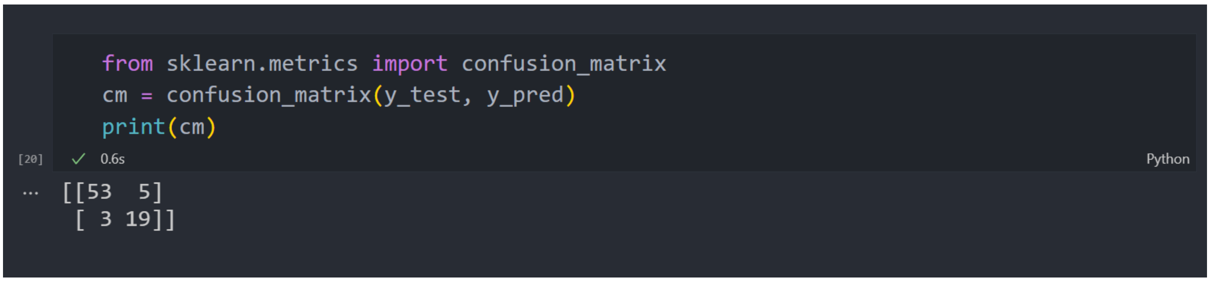
1.9. ຈົ່ງທົດສອບໂມເດວດ້ວຍການpredict(X_test).



```
y_pred = classifier.predict(X_test)
print(y_pred)
```

```
... [0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 1 1 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0
     0 0 1 0 0 0 0 1 0 0 1 0 1 1 0 0 1 0 1 0 0 1 0 0 1 0 1 0 1 0 0 0 1 1 0 0 1
     0 0 0 0 1 1]
```

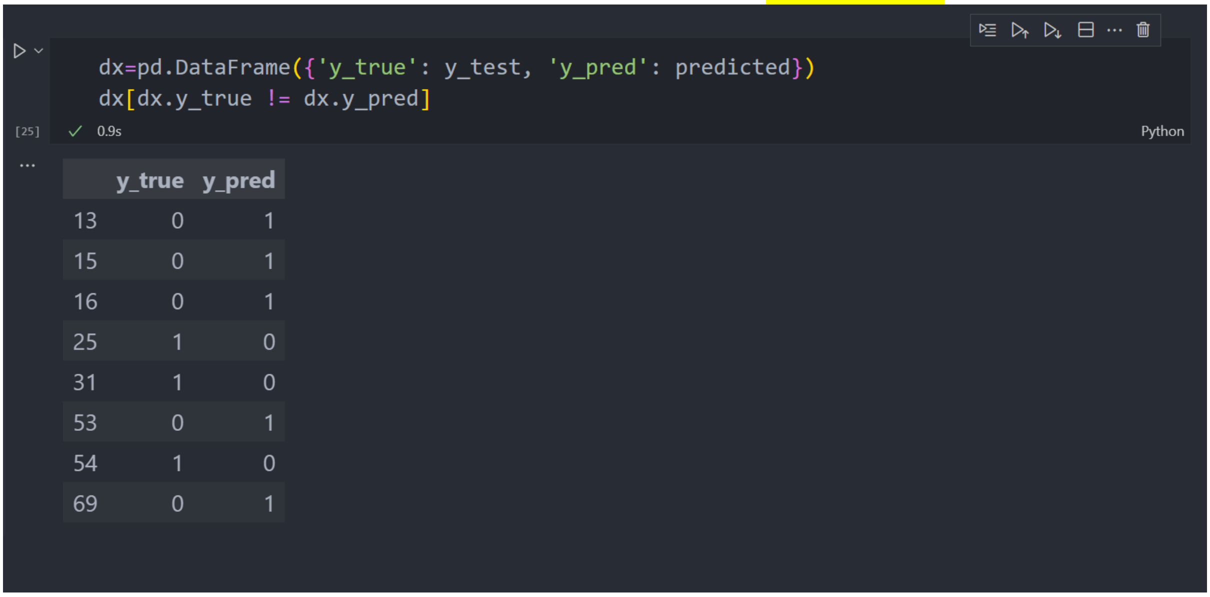
1.10. ຈົ່ງທຳການprocessing ດ້ວຍ confusion_matrix, ກຳນົດTP, TN, FP, FN



```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
... [[53  5]
     [ 3 19]]
```

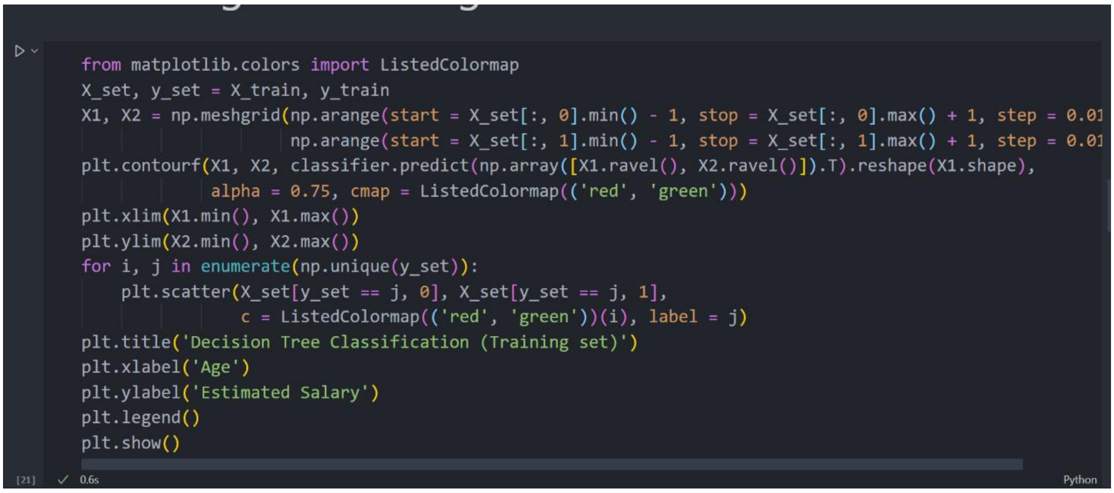
1.11. ຈົ່ງສະແດງຜົນດ້ວຍການສົມທຽບຄ່າຈິງ ແລະ ຄ່າຄາດເດົາຂອງ y_test ດ້ວຍມູບDataFrame



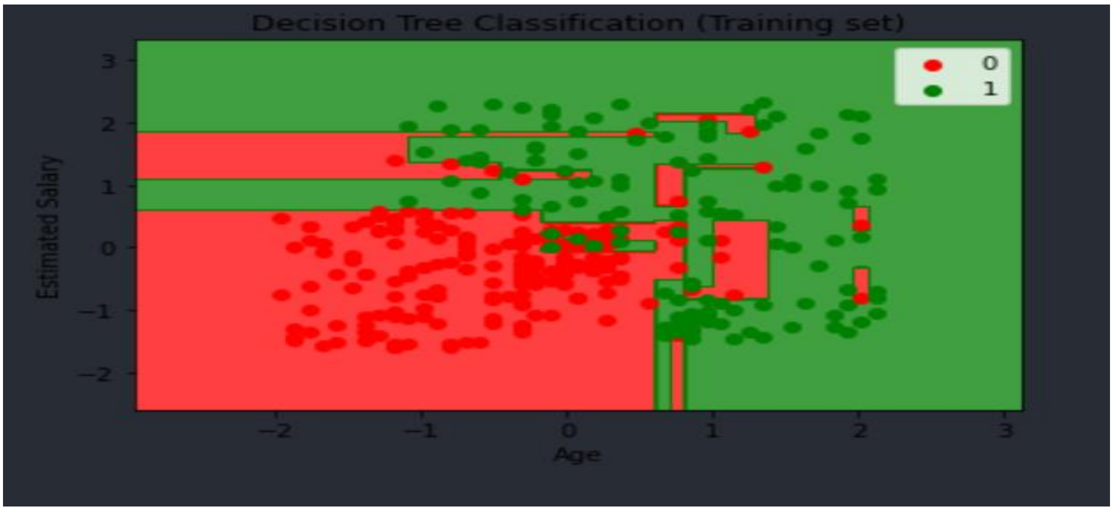
```
dx=pd.DataFrame({'y_true': y_test, 'y_pred': predicted})
dx[dx.y_true != dx.y_pred]
```

```
    y_true  y_pred
13       0       1
15       0       1
16       0       1
25       1       0
31       1       0
53       0       1
54       1       0
69       0       1
```

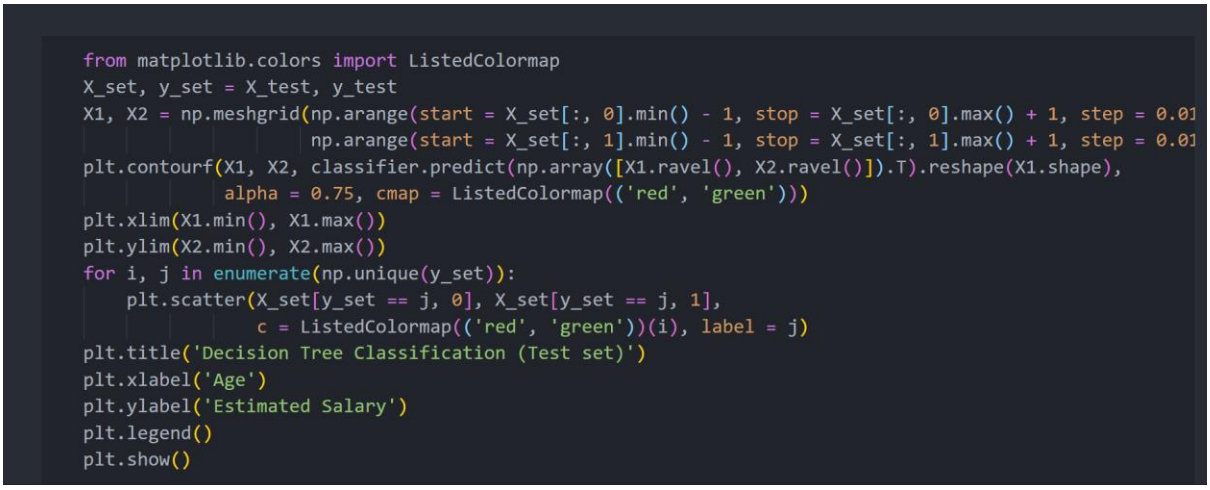
1. 12.ຈົ່ງສະແດງຂໍ້ມູນຊຸດຮຽນ (X_train, y_train) ດ້ວຍGraph ບົນພື້ນຖານຊຸດຄຳສັ່ງ matplotlib.



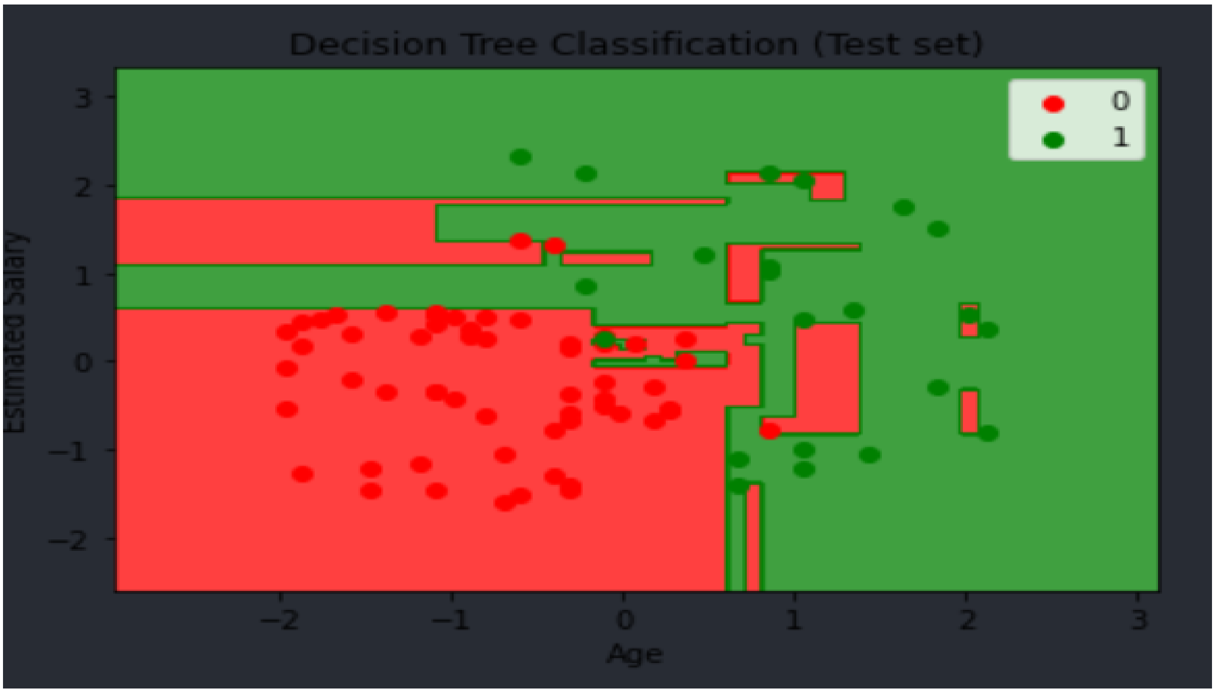
```
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('Decision Tree Classification (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```



1. 13.ຈົ່ງສະແດງຂໍ້ມູນຊຸດຮຽນ (X_test, y_test) ດ້ວຍGraph ບົນພື້ນຖານຊຸດຄຳສັ່ງ matplotlib.

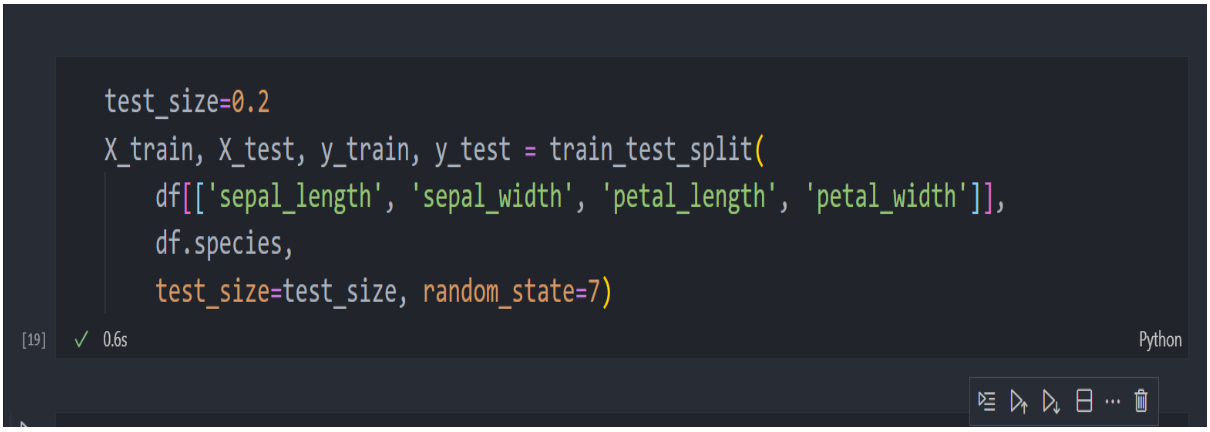


```
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('Decision Tree Classification (Test set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```



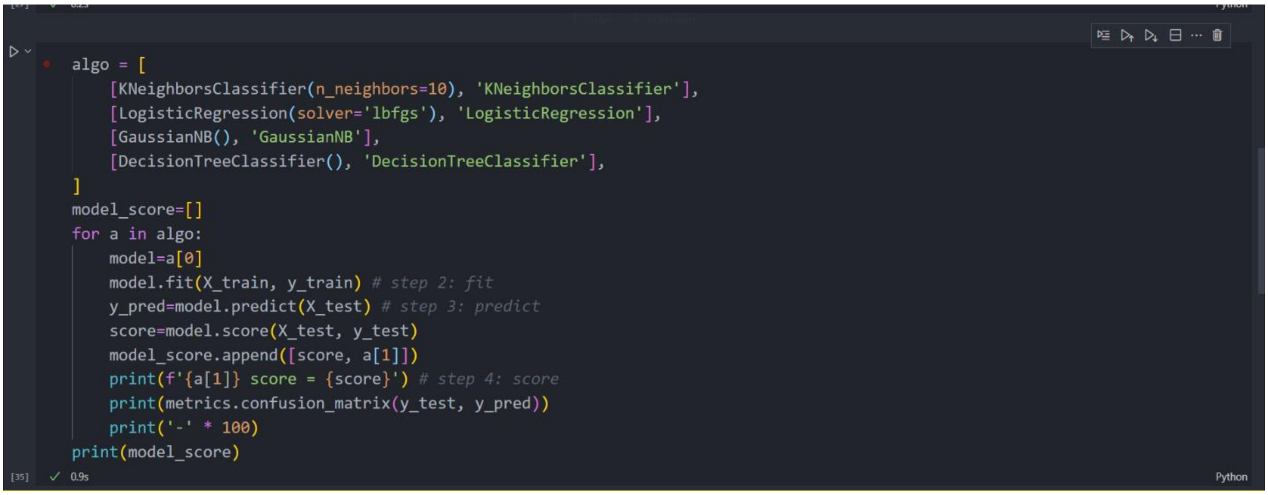
ພາກທີ 2

2.1 ຈາກຊຸດຂໍ້ມູນ iris.csv, ຈົ່ງເລືອກ sepal_length', 'sepal_width', 'petal_length', 'petal_width ເປັນ Features (X_train, X_test) ແລະ ໃຫ້ species ເປັນ Label (y_train, y_test), ແລ້ວແບ່ງຊຸດຮຽນ 80 ແລະ ຊຸດທົດສອບ 20 ?



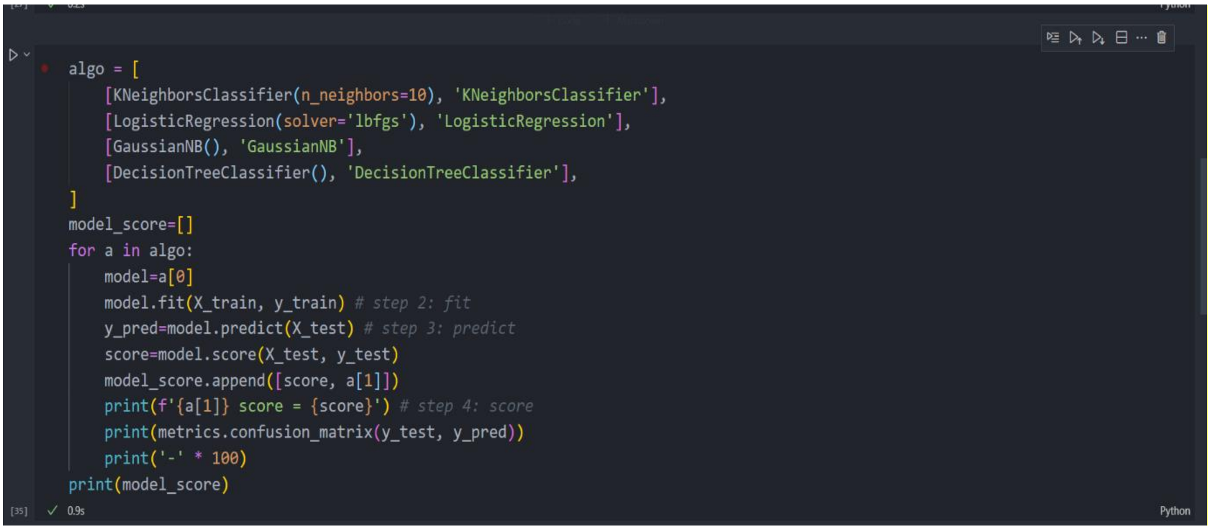
```
test_size=0.2
X_train, X_test, y_train, y_test = train_test_split(
    df[['sepal_length', 'sepal_width', 'petal_length', 'petal_width']],
    df.species,
    test_size=test_size, random_state=7)
```

2.2. ຈົ່ງສ້າງClassifier models ເຊັ່ນ: Kneighbors, Logistic Regression, Naive Bayes ແລະ DecisionTree ແລ້ວທຳການປະມວນຜົນ (fit) ຊຸດຂໍ້ມູນ (X_train, y_train)

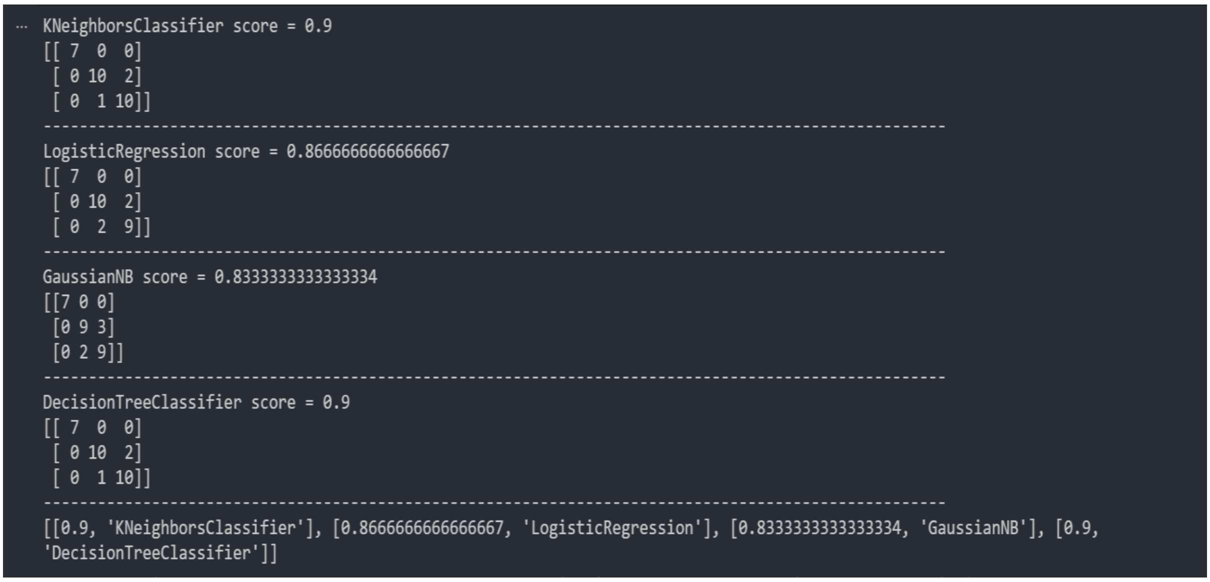


```
algo = [
    [KNeighborsClassifier(n_neighbors=10), 'KNeighborsClassifier'],
    [LogisticRegression(solver='lbfgs'), 'LogisticRegression'],
    [GaussianNB(), 'GaussianNB'],
    [DecisionTreeClassifier(), 'DecisionTreeClassifier'],
]
model_score=[]
for a in algo:
    model=a[0]
    model.fit(X_train, y_train) # step 2: fit
    y_pred=model.predict(X_test) # step 3: predict
    score=model.score(X_test, y_test)
    model_score.append([score, a[1]])
    print(f'{a[1]} score = {score}') # step 4: score
    print(metrics.confusion_matrix(y_test, y_pred))
    print('-' * 100)
print(model_score)
```

2.3. ຈົ່ງທົດສອບໂມເດວດ້ວຍການpredict(X_test) ແລະ ສົມທຽບປະສິດທິພາບຂອງບັນດາ models ດ້ວຍ confusion_matrix.



```python
algo = [
    [KNeighborsClassifier(n_neighbors=10), 'KNeighborsClassifier'],
    [LogisticRegression(solver='lbfgs'), 'LogisticRegression'],
    [GaussianNB(), 'GaussianNB'],
    [DecisionTreeClassifier(), 'DecisionTreeClassifier'],
]
model_score=[]
for a in algo:
    model=a[0]
    model.fit(X_train, y_train) # step 2: fit
    y_pred=model.predict(X_test) # step 3: predict
    score=model.score(X_test, y_test)
    model_score.append([score, a[1]])
    print(f'{a[1]} score = {score}') # step 4: score
    print(metrics.confusion_matrix(y_test, y_pred))
    print('-' * 100)
print(model_score)
```



```
KNeighborsClassifier score = 0.9
[[ 7  0  0]
 [ 0 10  2]
 [ 0  1 10]]
----------------------------------------------------------------------------------------------------
LogisticRegression score = 0.8666666666666667
[[ 7  0  0]
 [ 0 10  2]
 [ 0  2  9]]
----------------------------------------------------------------------------------------------------
GaussianNB score = 0.8333333333333334
[[7 0 0]
 [0 9 3]
 [0 2 9]]
----------------------------------------------------------------------------------------------------
DecisionTreeClassifier score = 0.9
[[ 7  0  0]
 [ 0 10  2]
 [ 0  1 10]]
----------------------------------------------------------------------------------------------------
[[0.9, 'KNeighborsClassifier'], [0.8666666666666667, 'LogisticRegression'], [0.8333333333333334, 'GaussianNB'], [0.9,
'DecisionTreeClassifier']]
```