PRAKTIKUM DATA MINING

Nama : Ahmad Izza Zain Firdaus

NIM : 19051214063

Kelas/Angkatan : SIB/2019

Algoritma : K-Nearest Neighbor (KNN)

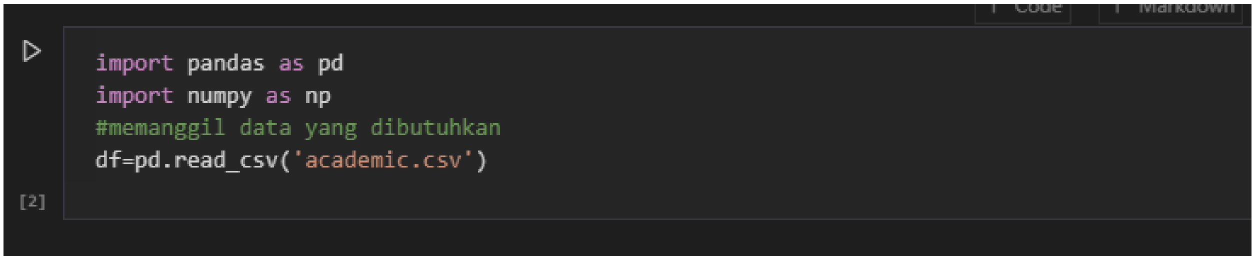
Jenis Analisis : Association

Dataset : academic.csv

Keterangan Dataset : dataset berisikan data pelajar-mahasiswa meliputi dari informasi diri hingga perilaku dalam pembelajaran dan dinilai terhadap keaktifan pelajar

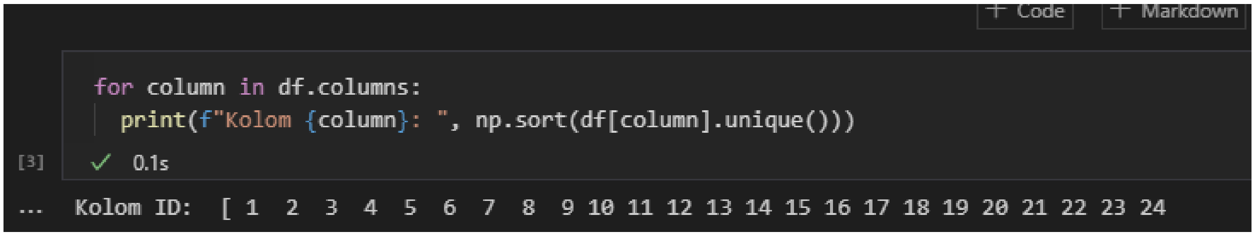
Metode Preprocessing : metode preprocessing dibagi menjadi beberapa tahapan setelah melakukan import data dan memanggil library yang dibutuhkan, dilakukan pengecekan isian dari masing-masing kolom untuk dianalisa

1. Dilakukan Import data untuk digunakan



```
import pandas as pd
import numpy as np
#memanggil data yang dibutuhkan
df=pd.read_csv('academic.csv')
```

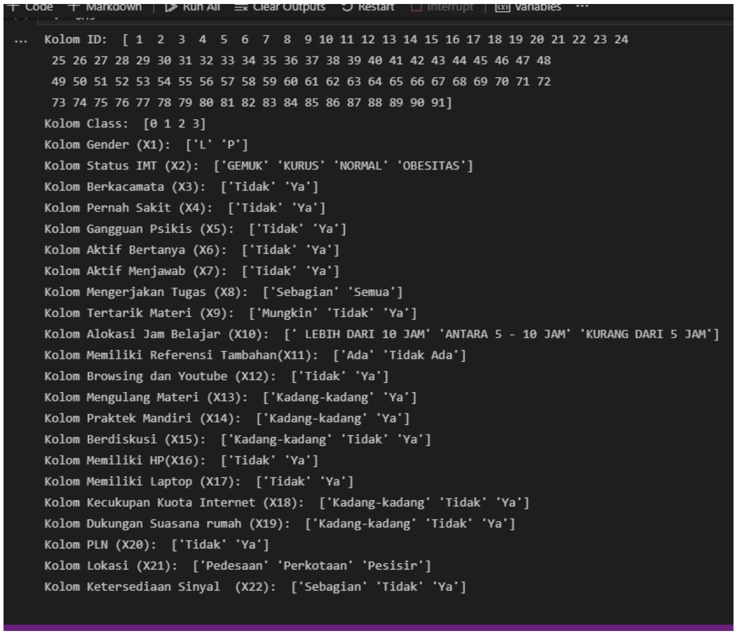
2. Mengecek jenis isian dari masing-masing kolom karena masih dalam bentuk data kategorik



```
for column in df.columns:
  print(f"Kolom {column}: ", np.sort(df[column].unique()))
```

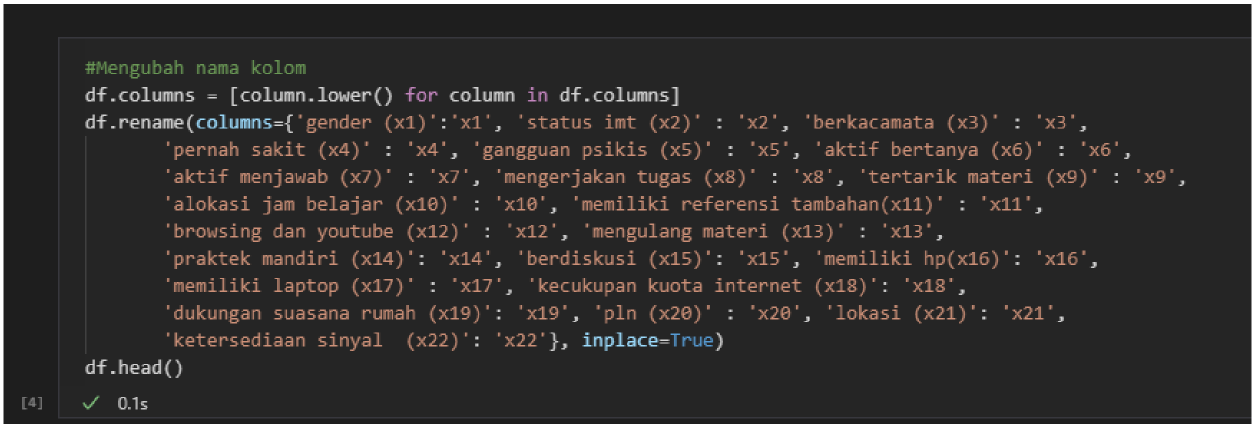
```
... Kolom ID:  [ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
```

Didapati hasil isian kolom sebagai berikut:



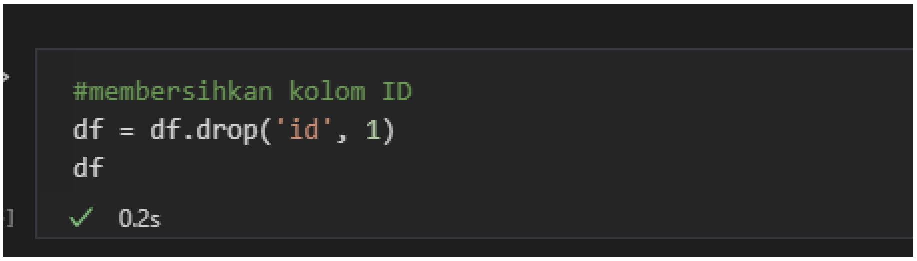
```
...  Kolom ID:  [ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
      25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
      49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72
      73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91]
     Kolom Class:  [0 1 2 3]
     Kolom Gender (X1):  ['L' 'P']
     Kolom Status IMT (X2):  ['GEMUK' 'KURUS' 'NORMAL' 'OBESITAS']
     Kolom Berkacamata (X3):  ['Tidak' 'Ya']
     Kolom Pernah Sakit (X4):  ['Tidak' 'Ya']
     Kolom Gangguan Psikis (X5):  ['Tidak' 'Ya']
     Kolom Aktif Bertanya (X6):  ['Tidak' 'Ya']
     Kolom Aktif Menjawab (X7):  ['Tidak' 'Ya']
     Kolom Mengerjakan Tugas (X8):  ['Sebagian' 'Semua']
     Kolom Tertarik Materi (X9):  ['Mungkin' 'Tidak' 'Ya']
     Kolom Alokasi Jam Belajar (X10):  [' LEBIH DARI 10 JAM' 'ANTARA 5 - 10 JAM' 'KURANG DARI 5 JAM']
     Kolom Memiliki Referensi Tambahan(X11):  ['Ada' 'Tidak Ada']
     Kolom Browsing dan Youtube (X12):  ['Tidak' 'Ya']
     Kolom Mengulang Materi (X13):  ['Kadang-kadang' 'Ya']
     Kolom Praktek Mandiri (X14):  ['Kadang-kadang' 'Ya']
     Kolom Berdiskusi (X15):  ['Kadang-kadang' 'Tidak' 'Ya']
     Kolom Memiliki HP(X16):  ['Tidak' 'Ya']
     Kolom Memiliki Laptop (X17):  ['Tidak' 'Ya']
     Kolom Kecukupan Kuota Internet (X18):  ['Kadang-kadang' 'Tidak' 'Ya']
     Kolom Dukungan Suasana rumah (X19):  ['Kadang-kadang' 'Tidak' 'Ya']
     Kolom PLN (X20):  ['Tidak' 'Ya']
     Kolom Lokasi (X21):  ['Pedesaan' 'Perkotaan' 'Pesisir']
     Kolom Ketersediaan Sinyal  (X22):  ['Sebagian' 'Tidak' 'Ya']
```

3. Setelah mendapatkan isian dari masing-masing kolom, karena penamaan kolom terbilang rumit, maka dilakukan pengubahan nama kolom untuk memudahkan proses berikutnya



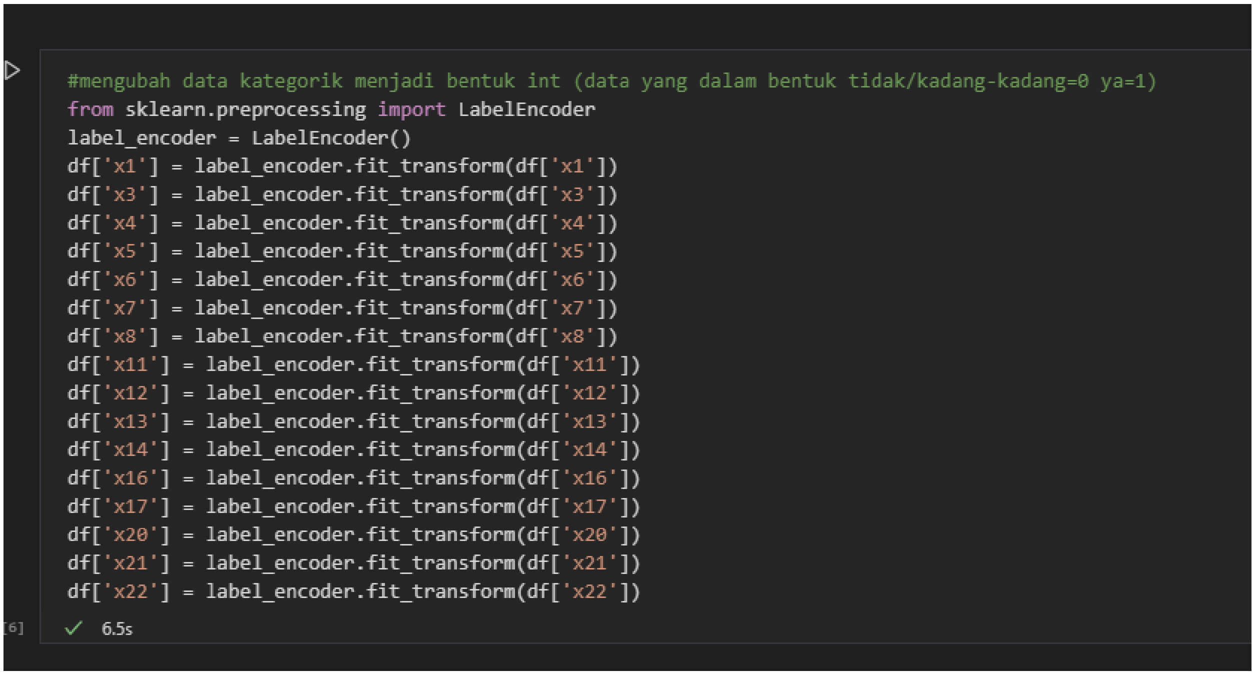
```
#Mengubah nama kolom
df.columns = [column.lower() for column in df.columns]
df.rename(columns={'gender (x1)':'x1', 'status imt (x2)' : 'x2', 'berkacamata (x3)' : 'x3',
          'pernah sakit (x4)' : 'x4', 'gangguan psikis (x5)' : 'x5', 'aktif bertanya (x6)' : 'x6',
          'aktif menjawab (x7)' : 'x7', 'mengerjakan tugas (x8)' : 'x8', 'tertarik materi (x9)' : 'x9',
          'alokasi jam belajar (x10)' : 'x10', 'memiliki referensi tambahan(x11)' : 'x11',
          'browsing dan youtube (x12)' : 'x12', 'mengulang materi (x13)' : 'x13',
          'praktek mandiri (x14)': 'x14', 'berdiskusi (x15)': 'x15', 'memiliki hp(x16)': 'x16',
          'memiliki laptop (x17)' : 'x17', 'kecukupan kuota internet (x18)': 'x18',
          'dukungan suasana rumah (x19)': 'x19', 'pln (x20)' : 'x20', 'lokasi (x21)': 'x21',
          'ketersediaan sinyal  (x22)': 'x22'}, inplace=True)
df.head()
```

4. Berdasarkan point nomor 2, kolom id tidak diperlukan dalam proses sebagai variabel independen maupun dependen, oleh karena itu bisa dilakukan pengeluaran variabel



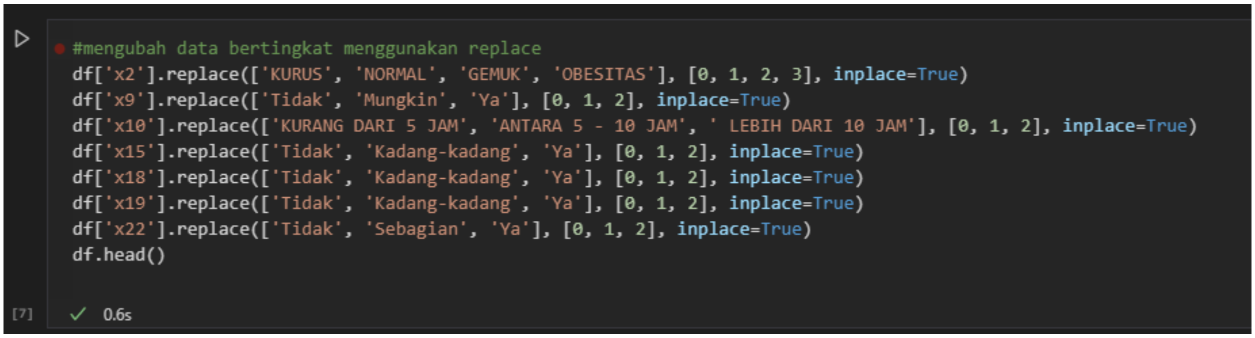
```
#membersihkan kolom ID
df = df.drop('id', 1)
df
```

5. Dilakukan pengubahan tipe data dari masing masing kolom, dimulai dari data kategorik yang bersifat nominal, diubah dengan bantuan fungsi preprocessing di sklearn yakni LabelEncoder(), semua data termasuk data nominal selain x2, x9, x10, x15, x18, x19, x22



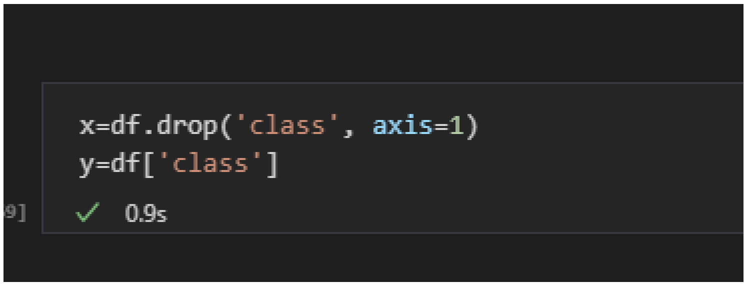
```
#mengubah data kategorik menjadi bentuk int (data yang dalam bentuk tidak/kadang-kadang=0 ya=1)
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
df['x1'] = label_encoder.fit_transform(df['x1'])
df['x3'] = label_encoder.fit_transform(df['x3'])
df['x4'] = label_encoder.fit_transform(df['x4'])
df['x5'] = label_encoder.fit_transform(df['x5'])
df['x6'] = label_encoder.fit_transform(df['x6'])
df['x7'] = label_encoder.fit_transform(df['x7'])
df['x8'] = label_encoder.fit_transform(df['x8'])
df['x11'] = label_encoder.fit_transform(df['x11'])
df['x12'] = label_encoder.fit_transform(df['x12'])
df['x13'] = label_encoder.fit_transform(df['x13'])
df['x14'] = label_encoder.fit_transform(df['x14'])
df['x16'] = label_encoder.fit_transform(df['x16'])
df['x17'] = label_encoder.fit_transform(df['x17'])
df['x20'] = label_encoder.fit_transform(df['x20'])
df['x21'] = label_encoder.fit_transform(df['x21'])
df['x22'] = label_encoder.fit_transform(df['x22'])
```

6. Sisa data uang bersifat ordinal diubah secara manual dengan menggunakan perintah replace



```
#mengubah data bertingkat menggunakan replace
df['x2'].replace(['KURUS', 'NORMAL', 'GEMUK', 'OBESITAS'], [0, 1, 2, 3], inplace=True)
df['x9'].replace(['Tidak', 'Mungkin', 'Ya'], [0, 1, 2], inplace=True)
df['x10'].replace(['KURANG DARI 5 JAM', 'ANTARA 5 - 10 JAM', ' LEBIH DARI 10 JAM'], [0, 1, 2], inplace=True)
df['x15'].replace(['Tidak', 'Kadang-kadang', 'Ya'], [0, 1, 2], inplace=True)
df['x18'].replace(['Tidak', 'Kadang-kadang', 'Ya'], [0, 1, 2], inplace=True)
df['x19'].replace(['Tidak', 'Kadang-kadang', 'Ya'], [0, 1, 2], inplace=True)
df['x22'].replace(['Tidak', 'Sebagian', 'Ya'], [0, 1, 2], inplace=True)
df.head()
```

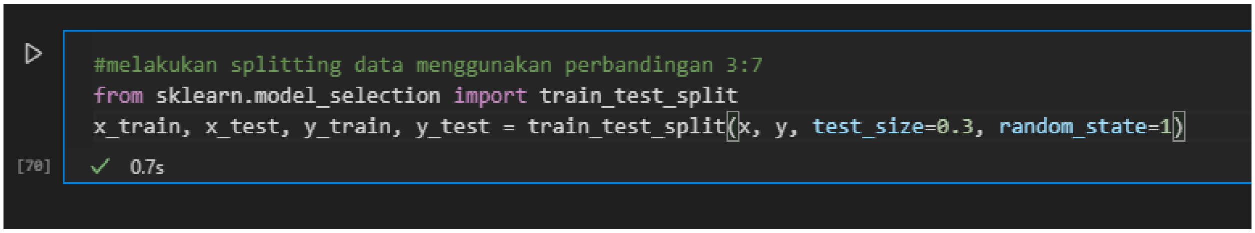
7. Setelah semua data diubah kedalam bentuk integer maka selanjutnya dilakukan penetuan data yang menjadi variabel dependen dan independen. Kolom class akan menjadi variabel dependen diwakili y dan selain itu menjadi variabel independen diwakili x



```
x=df.drop('class', axis=1)
y=df['class']
```

9] ✓ 0.9s

8. Setelah dimasukkan ke dalam variabel, selanjutnya dibagi menjadi nilai yang akan dijadikan train dan test dengan fungsi sklearn, untuk ukuran menggunakan pembagian 7:3

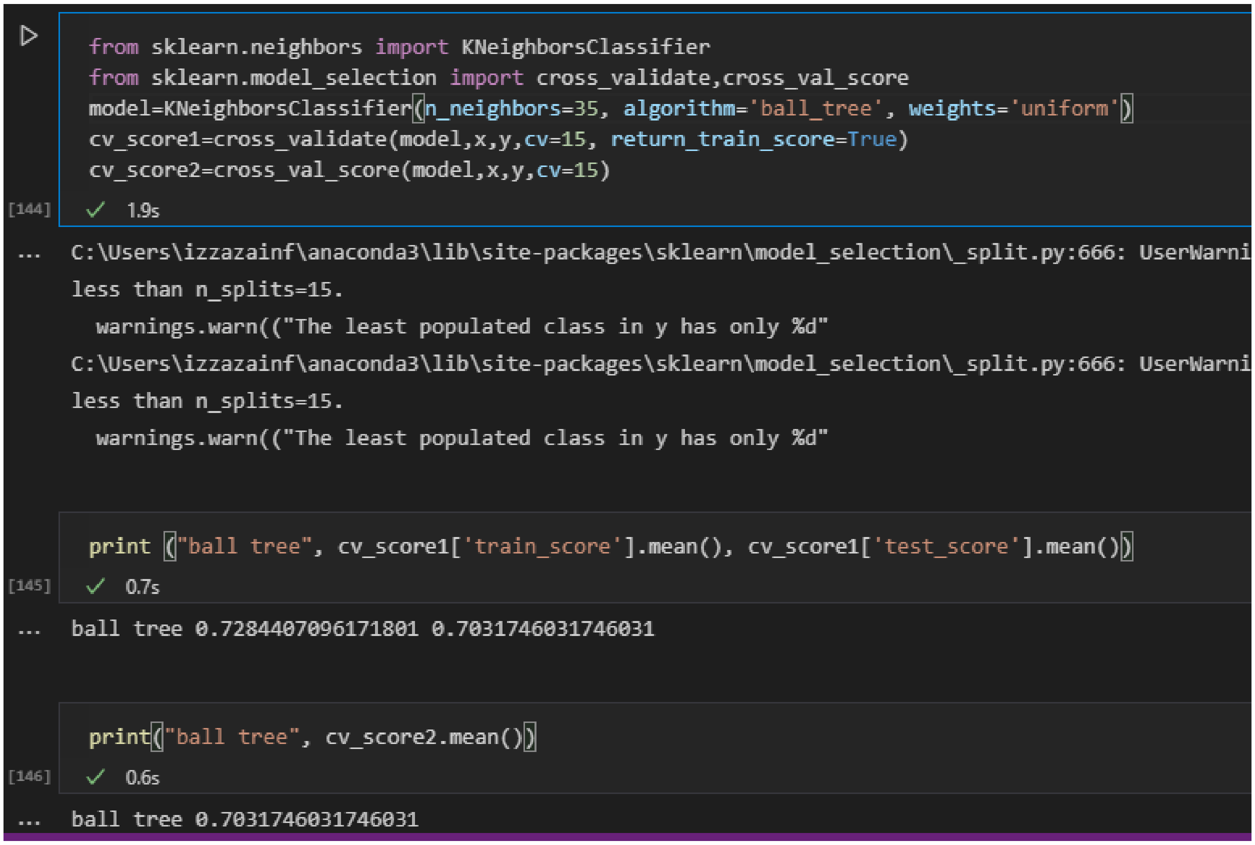


```
#melakukan splitting data menggunakan perbandingan 3:7
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=1)
```

[70] ✓ 0.7s

Pembahasan : hasil percobaan penghitungan menggunakan beberapa metode adalah sebagai berikut:

1. Penghitungan Manual
   a. Weight: Uniform, Algoritm: ball tree



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=35, algorithm='ball_tree', weights='uniform')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

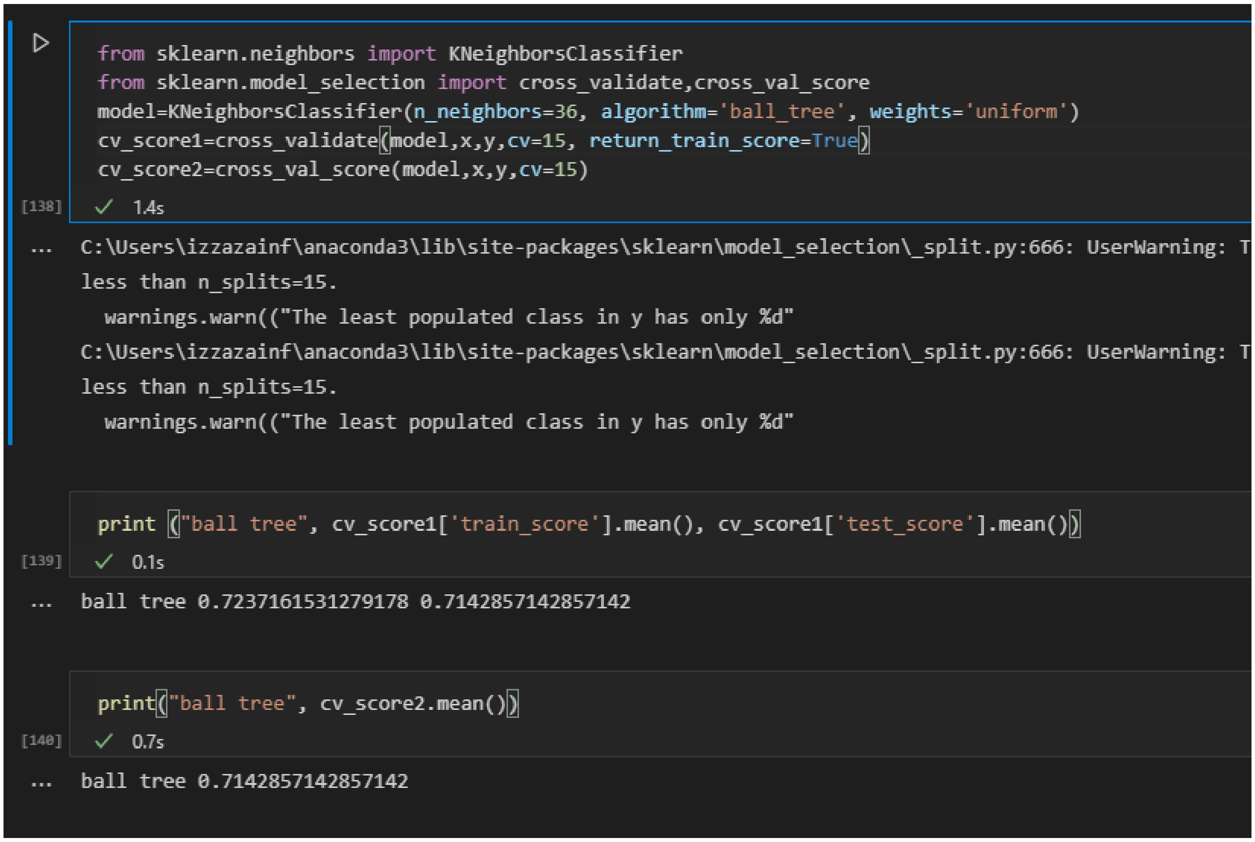
```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarni
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarni
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

```
print ("ball tree", cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
ball tree 0.7284407096171801 0.7031746031746031
```

```
print("ball tree", cv_score2.mean())
```

```
ball tree 0.7031746031746031
```



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=36, algorithm='ball_tree', weights='uniform')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

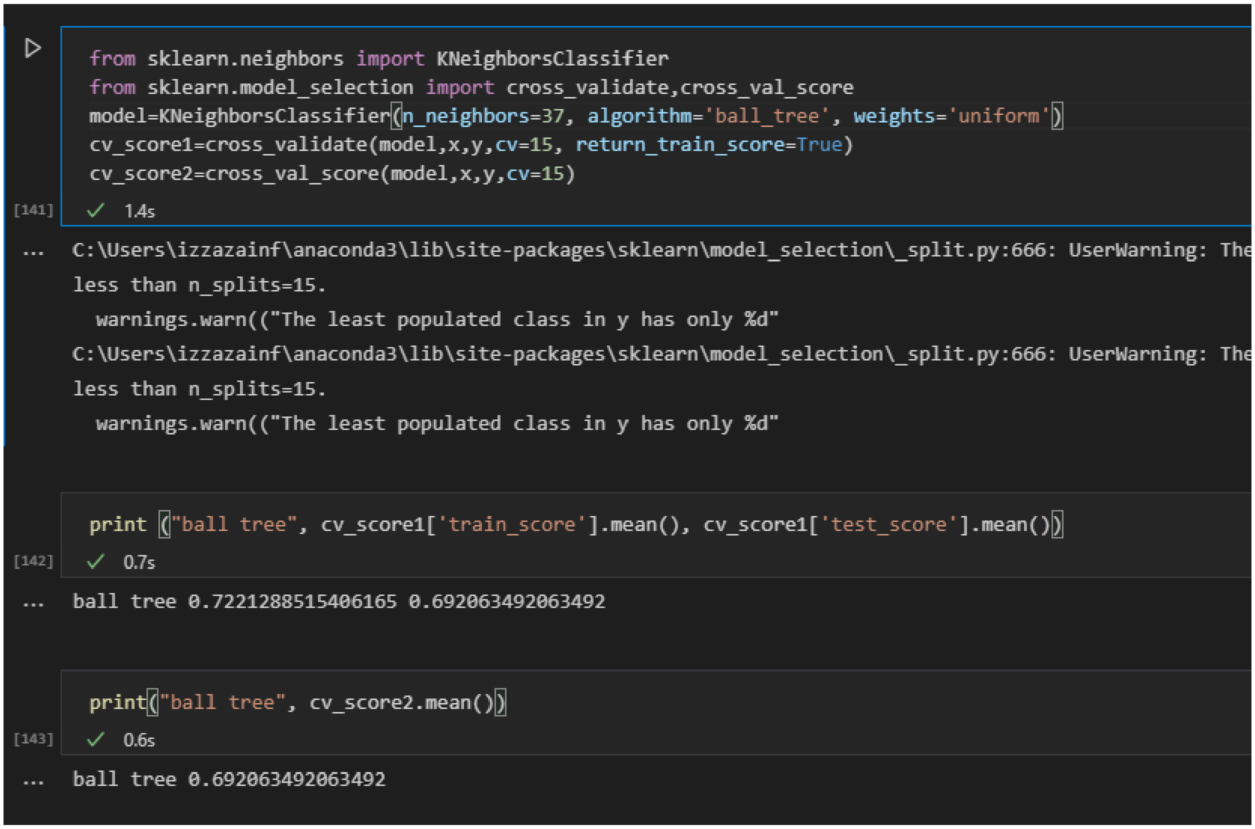
```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarning: T
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarning: T
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

```
print ("ball tree", cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
ball tree 0.7237161531279178 0.7142857142857142
```

```
print("ball tree", cv_score2.mean())
```

```
ball tree 0.7142857142857142
```



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=37, algorithm='ball_tree', weights='uniform')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarning: The
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarning: The
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

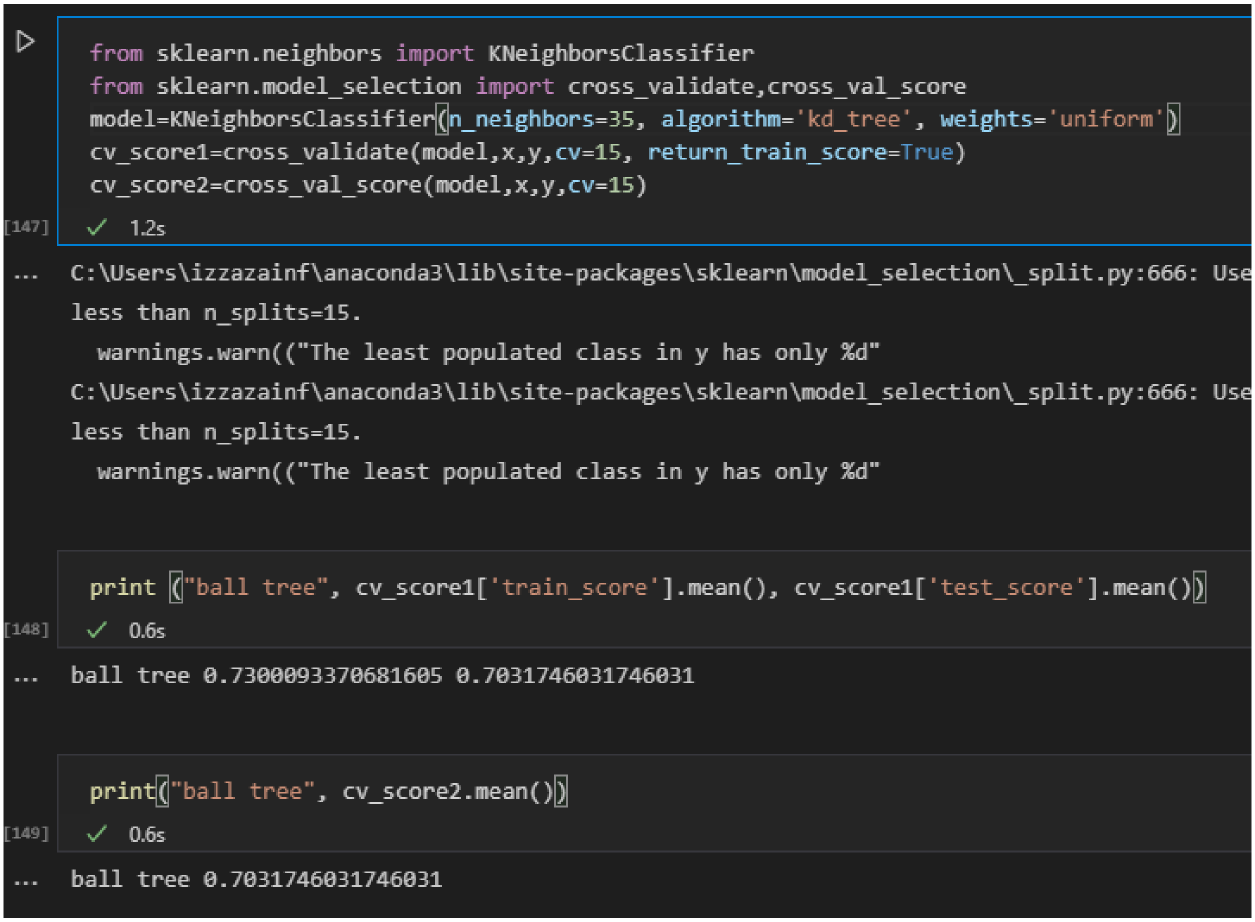
```
print ("ball tree", cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
ball tree 0.7221288515406165 0.692063492063492
```

```
print("ball tree", cv_score2.mean())
```

```
ball tree 0.692063492063492
```

b. Weight: Uniform, Algoritm: kd_tree



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=35, algorithm='kd_tree', weights='uniform')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

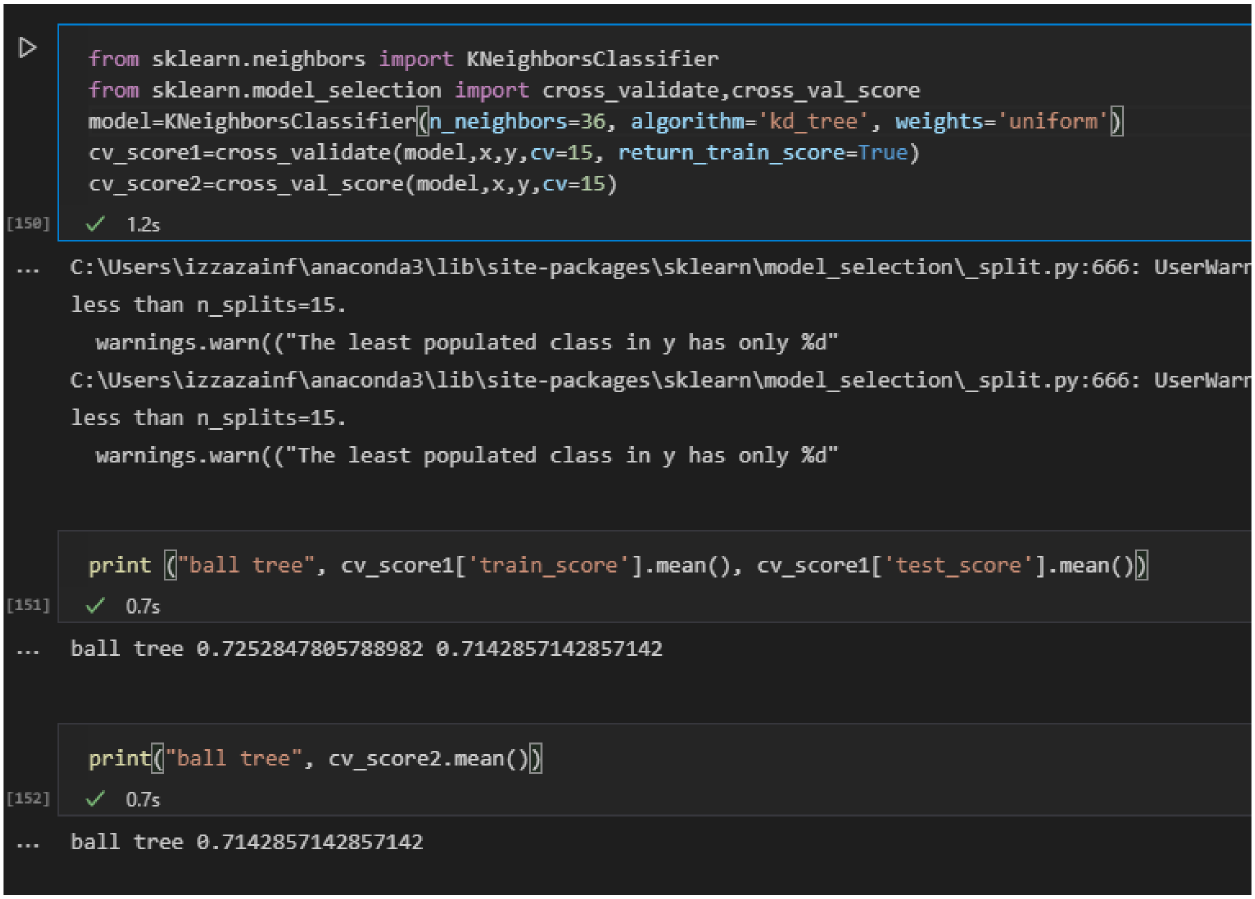
```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: Use
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: Use
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

```
print ("ball tree", cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
ball tree 0.7300093370681605 0.7031746031746031
```

```
print("ball tree", cv_score2.mean())
```

```
ball tree 0.7031746031746031
```



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=36, algorithm='kd_tree', weights='uniform')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

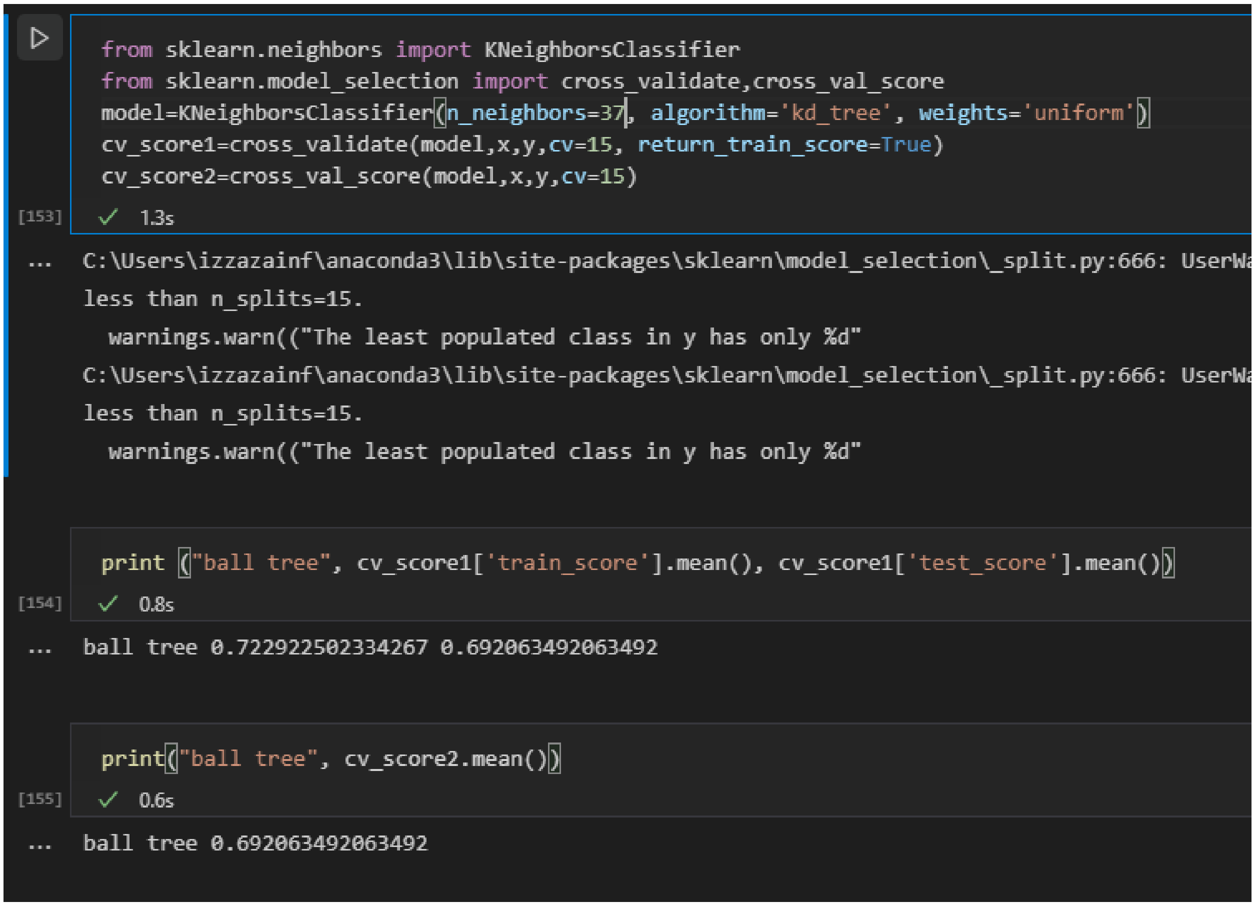
```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarn
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarn
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

```
print ("ball tree", cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
ball tree 0.7252847805788982 0.7142857142857142
```

```
print("ball tree", cv_score2.mean())
```

```
ball tree 0.7142857142857142
```



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=37, algorithm='kd_tree', weights='uniform')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWa
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWa
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

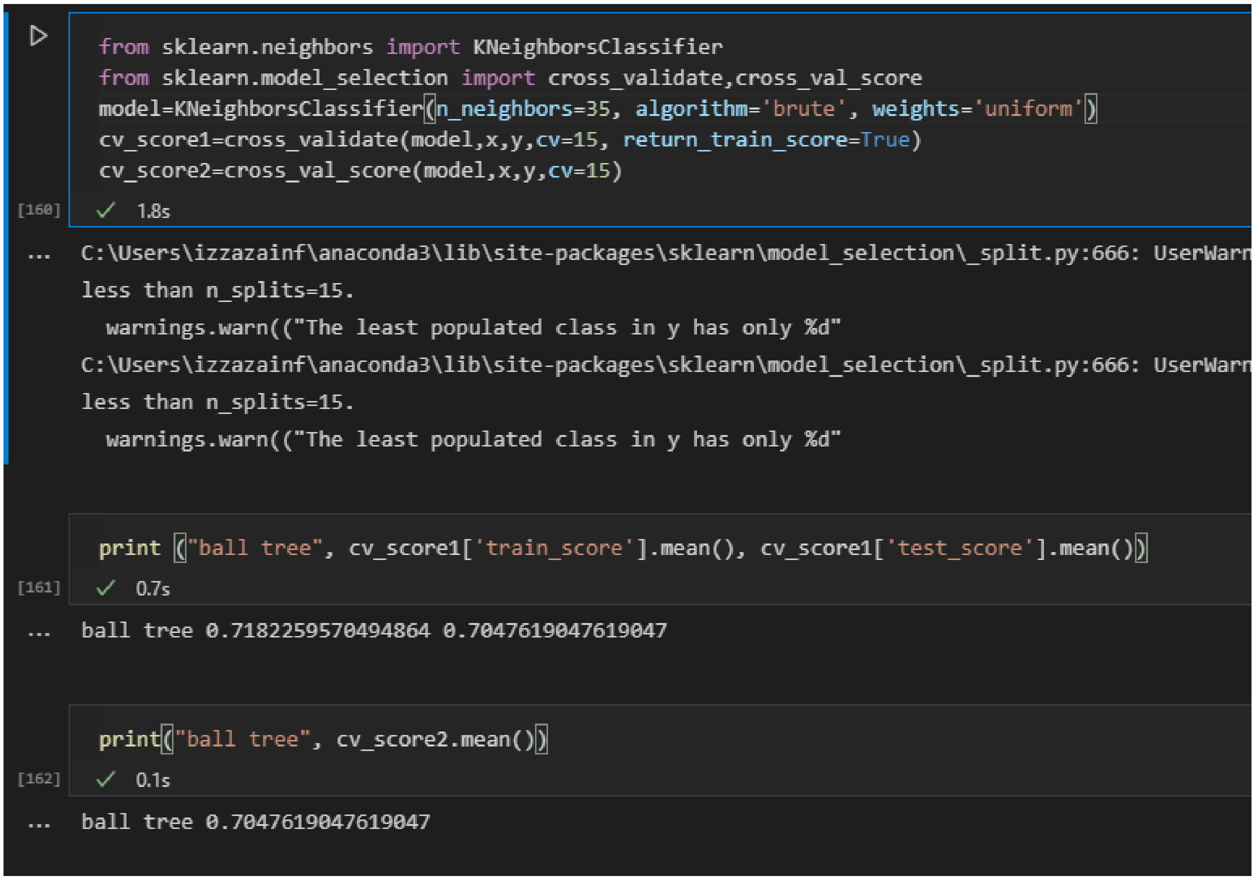
```
print ("ball tree", cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
ball tree 0.722922502334267 0.692063492063492
```

```
print("ball tree", cv_score2.mean())
```

```
ball tree 0.692063492063492
```

c. Weight: Uniform, Algoritm: brute



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=35, algorithm='brute', weights='uniform')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

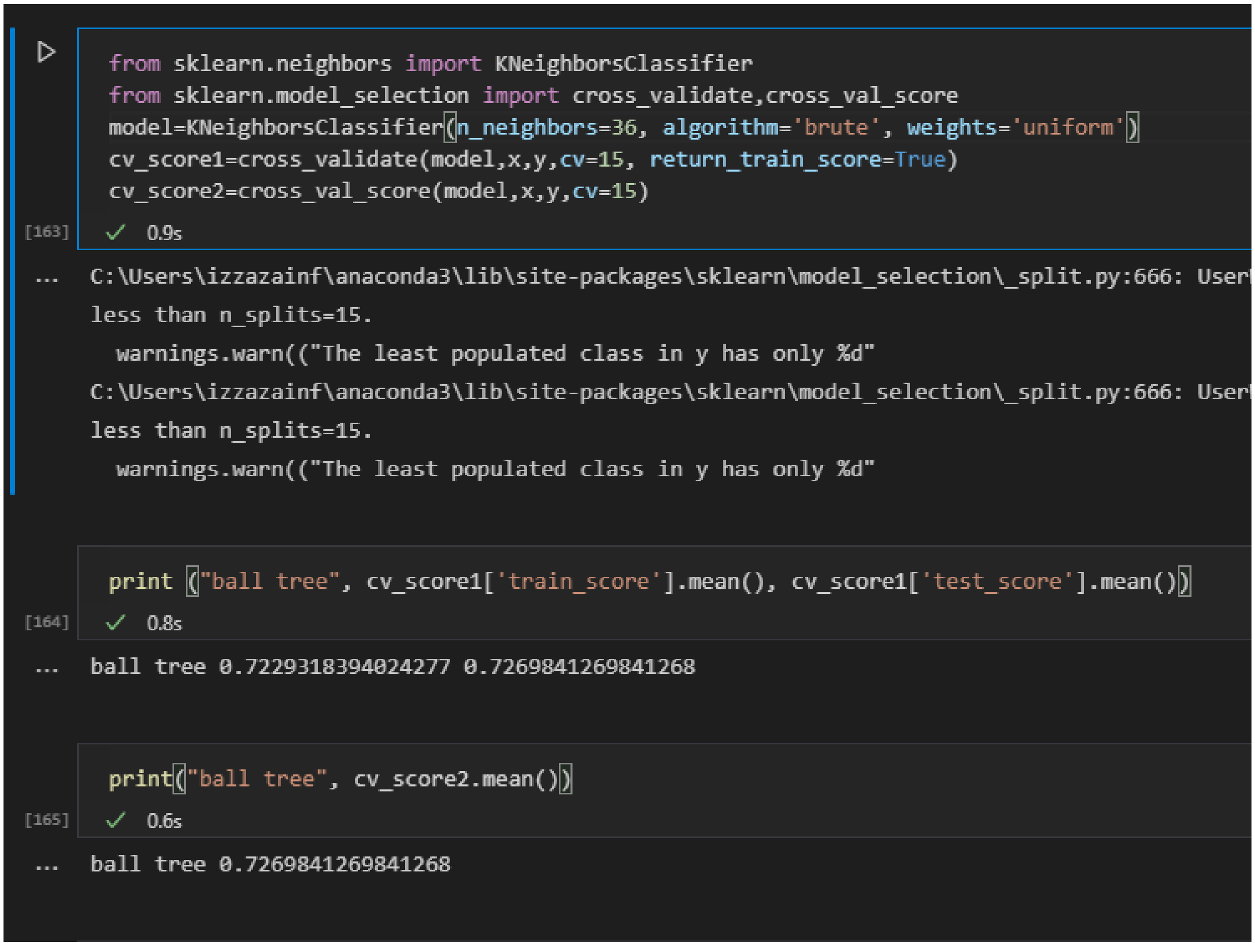
```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarn
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarn
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

```
print ("ball tree", cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
ball tree 0.7182259570494864 0.7047619047619047
```

```
print("ball tree", cv_score2.mean())
```

```
ball tree 0.7047619047619047
```



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=36, algorithm='brute', weights='uniform')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

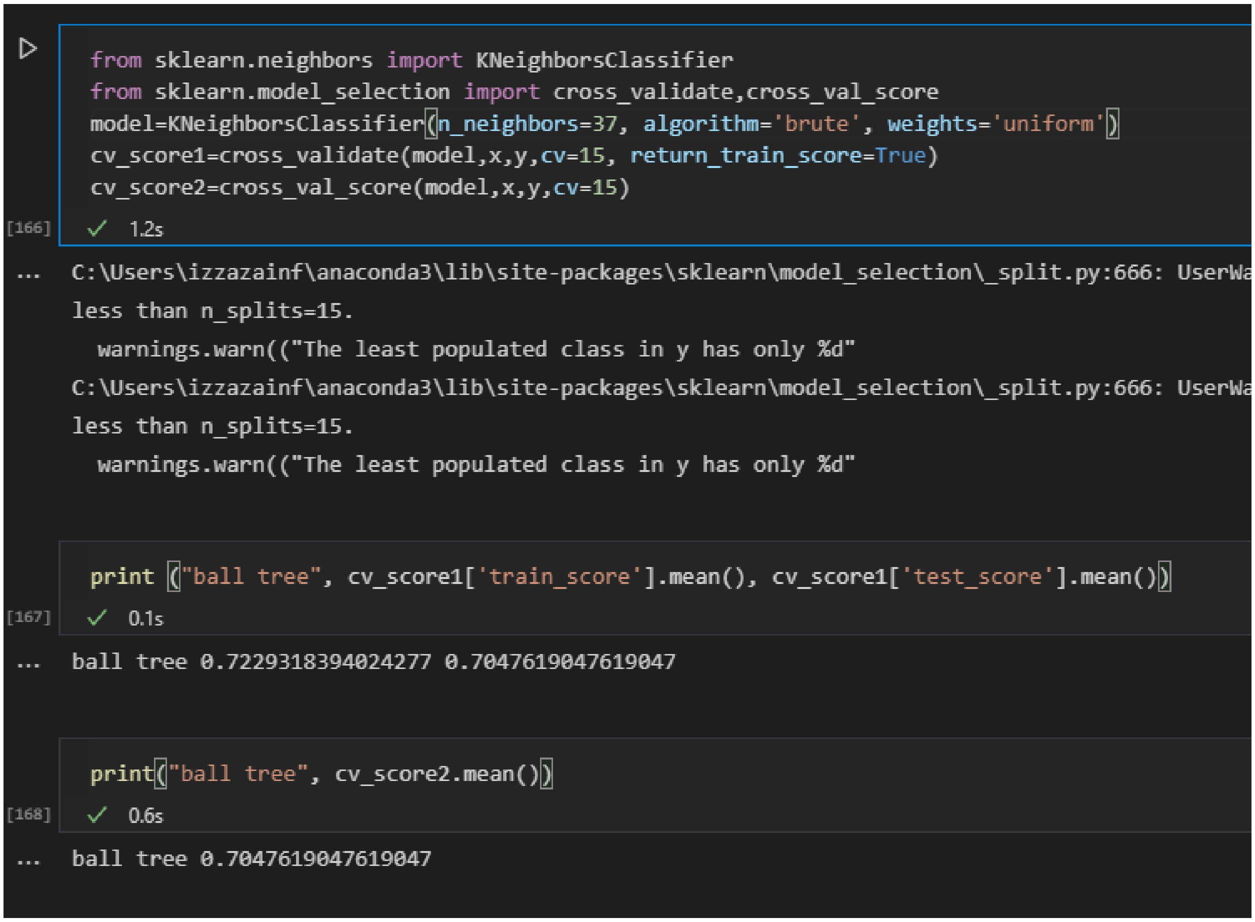
```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: User
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: User
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

```
print ("ball tree", cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
ball tree 0.7229318394024277 0.7269841269841268
```

```
print("ball tree", cv_score2.mean())
```

```
ball tree 0.7269841269841268
```



```python
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=37, algorithm='brute', weights='uniform')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWa
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWa
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

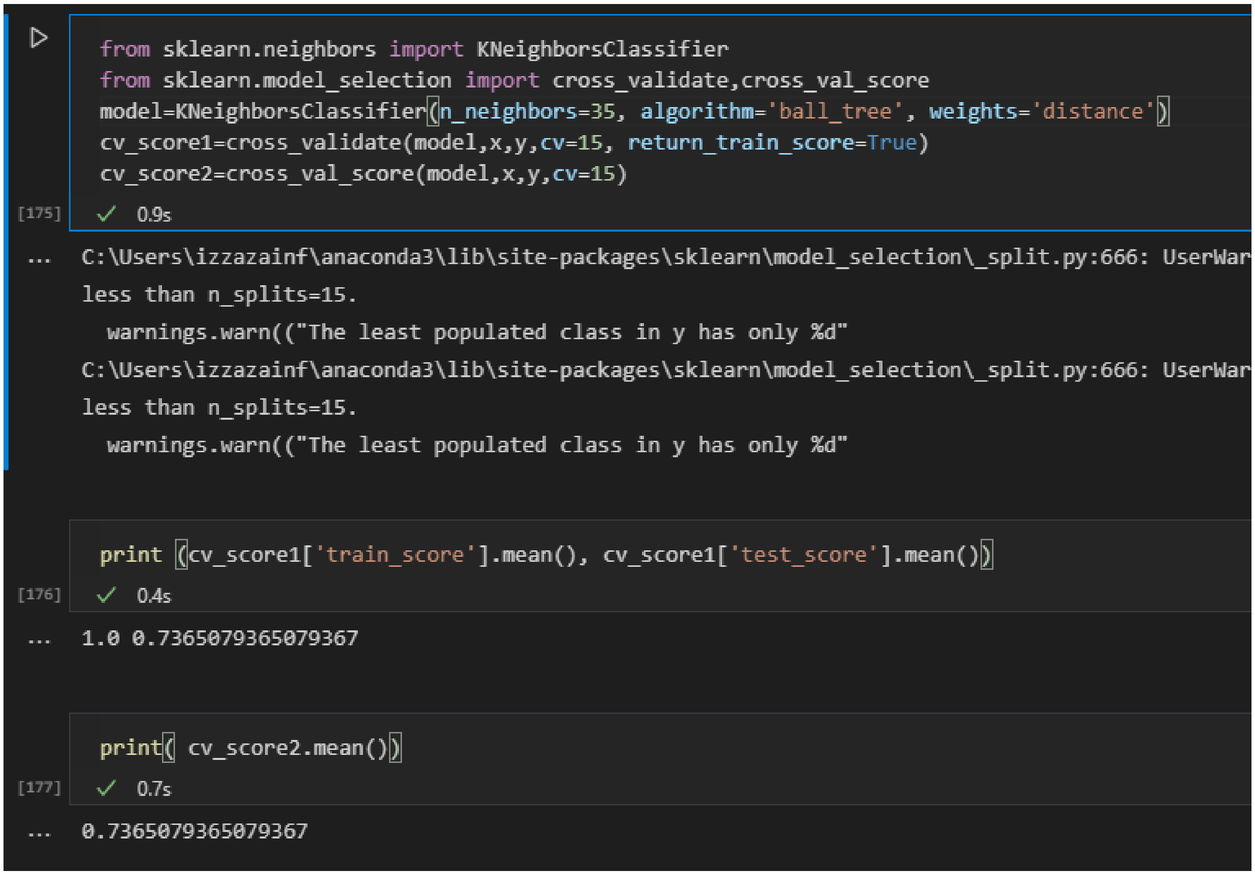
```python
print ("ball tree", cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
ball tree 0.7229318394024277 0.7047619047619047
```

```python
print("ball tree", cv_score2.mean())
```

```
ball tree 0.7047619047619047
```

d. Weight: distance, algorithm: ball_tree



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=35, algorithm='ball_tree', weights='distance')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

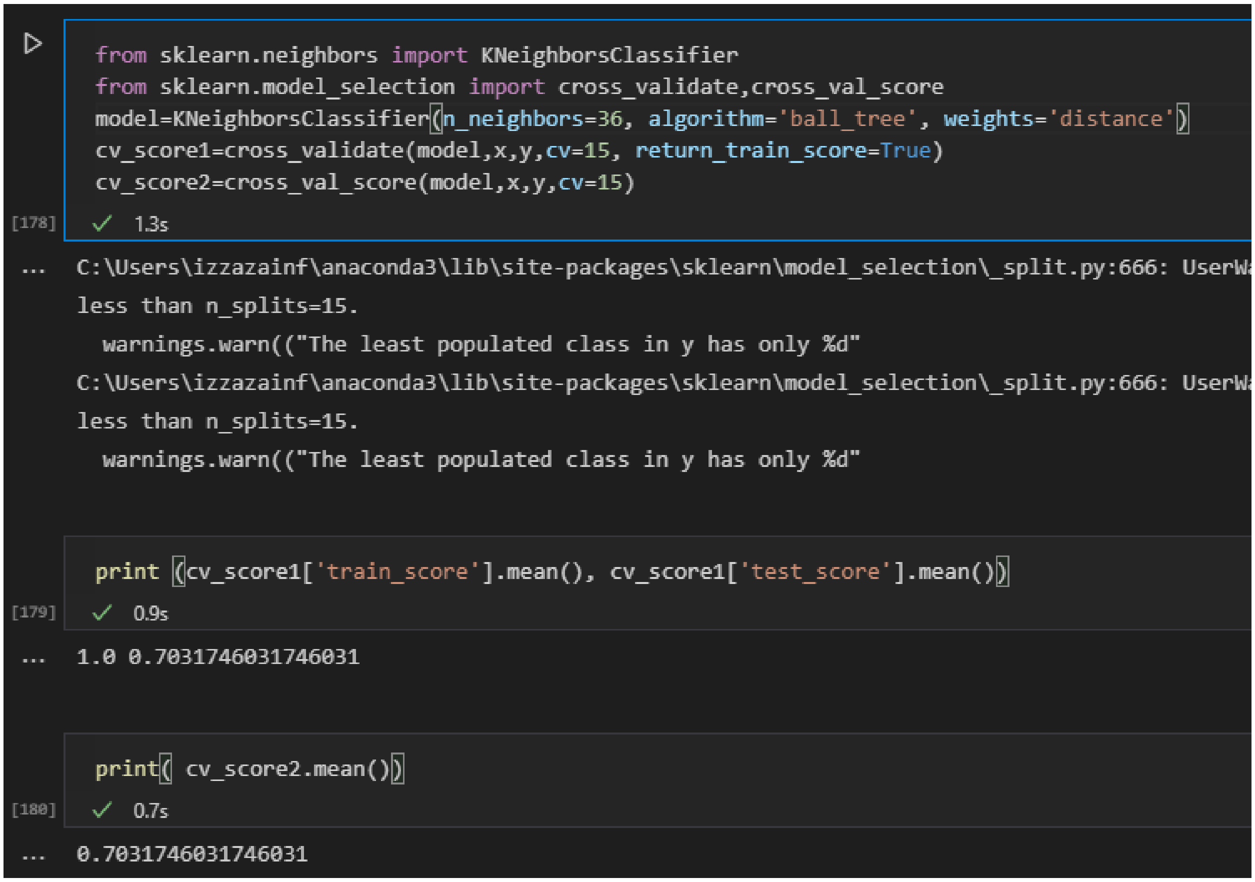
```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWar
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWar
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

```
print (cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
1.0 0.7365079365079367
```

```
print( cv_score2.mean())
```

```
0.7365079365079367
```



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=36, algorithm='ball_tree', weights='distance')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

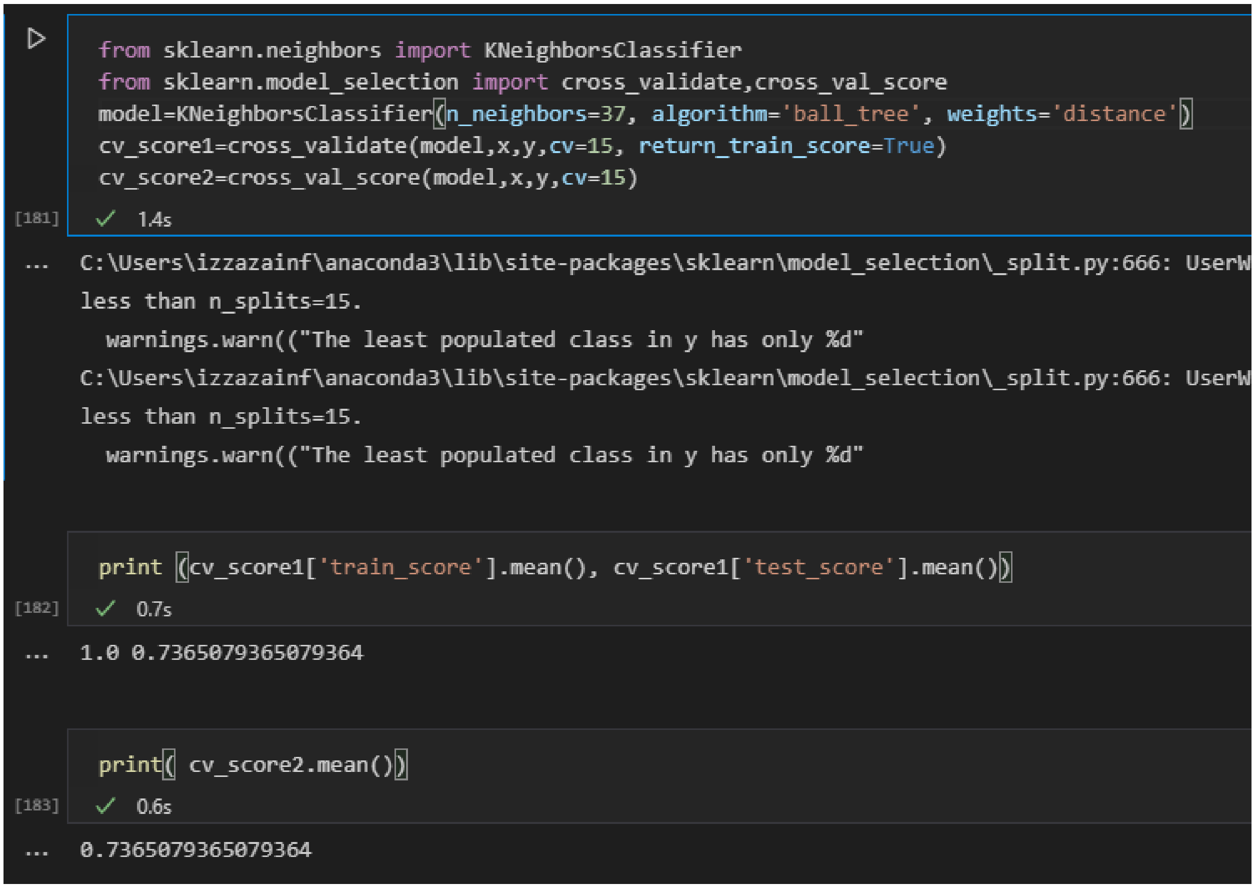
```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWa
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWa
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

```
print (cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
1.0 0.7031746031746031
```

```
print( cv_score2.mean())
```

```
0.7031746031746031
```



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=37, algorithm='ball_tree', weights='distance')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

[181] ✓ 1.4s

```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserW
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserW
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

```
print (cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

[182] ✓ 0.7s

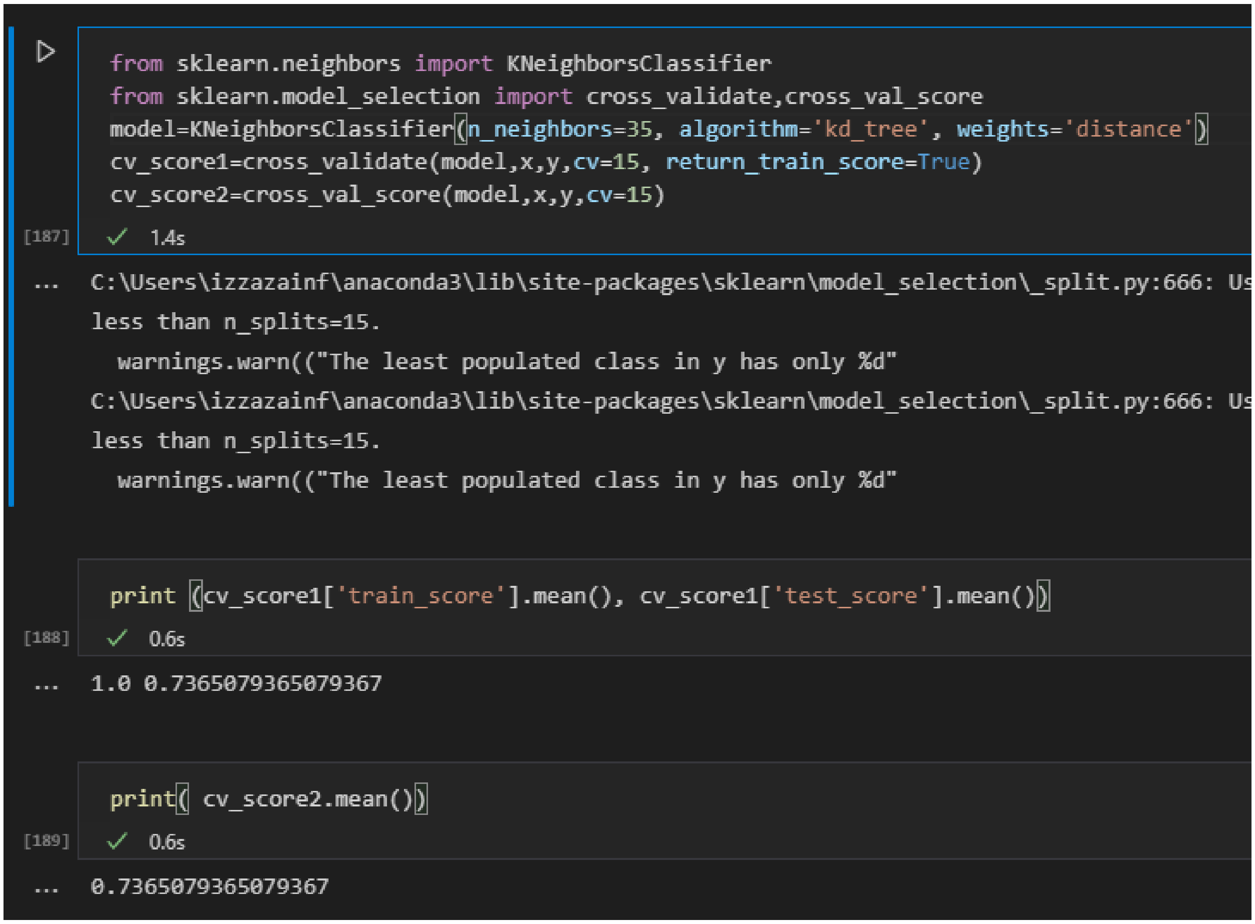
```
1.0 0.7365079365079364
```

```
print( cv_score2.mean())
```

[183] ✓ 0.6s

```
0.7365079365079364
```

e. Weight: distance, algorithm: kd_tree



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=35, algorithm='kd_tree', weights='distance')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

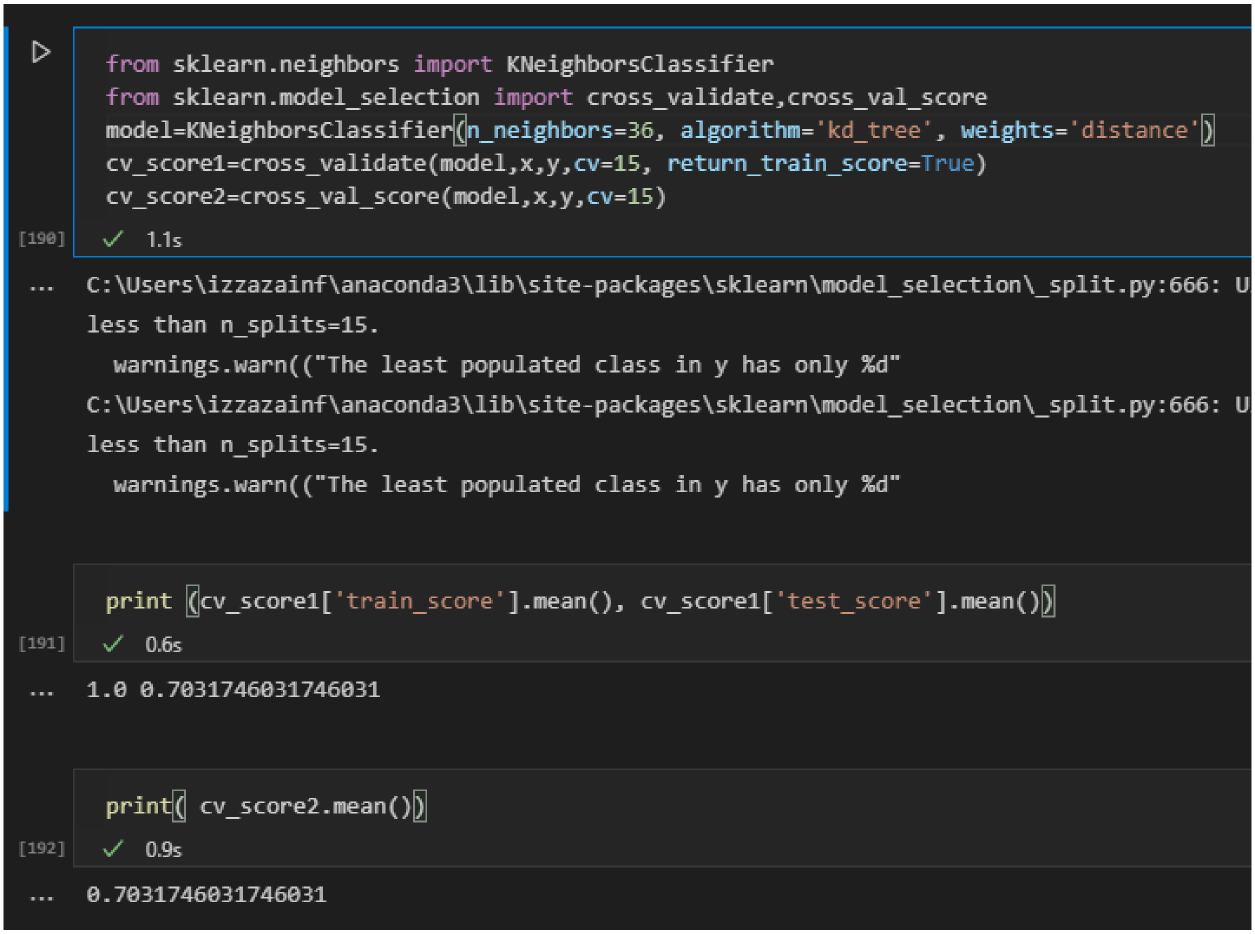
```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: Us
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: Us
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

```
print (cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
1.0 0.7365079365079367
```

```
print( cv_score2.mean())
```

```
0.7365079365079367
```



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=36, algorithm='kd_tree', weights='distance')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

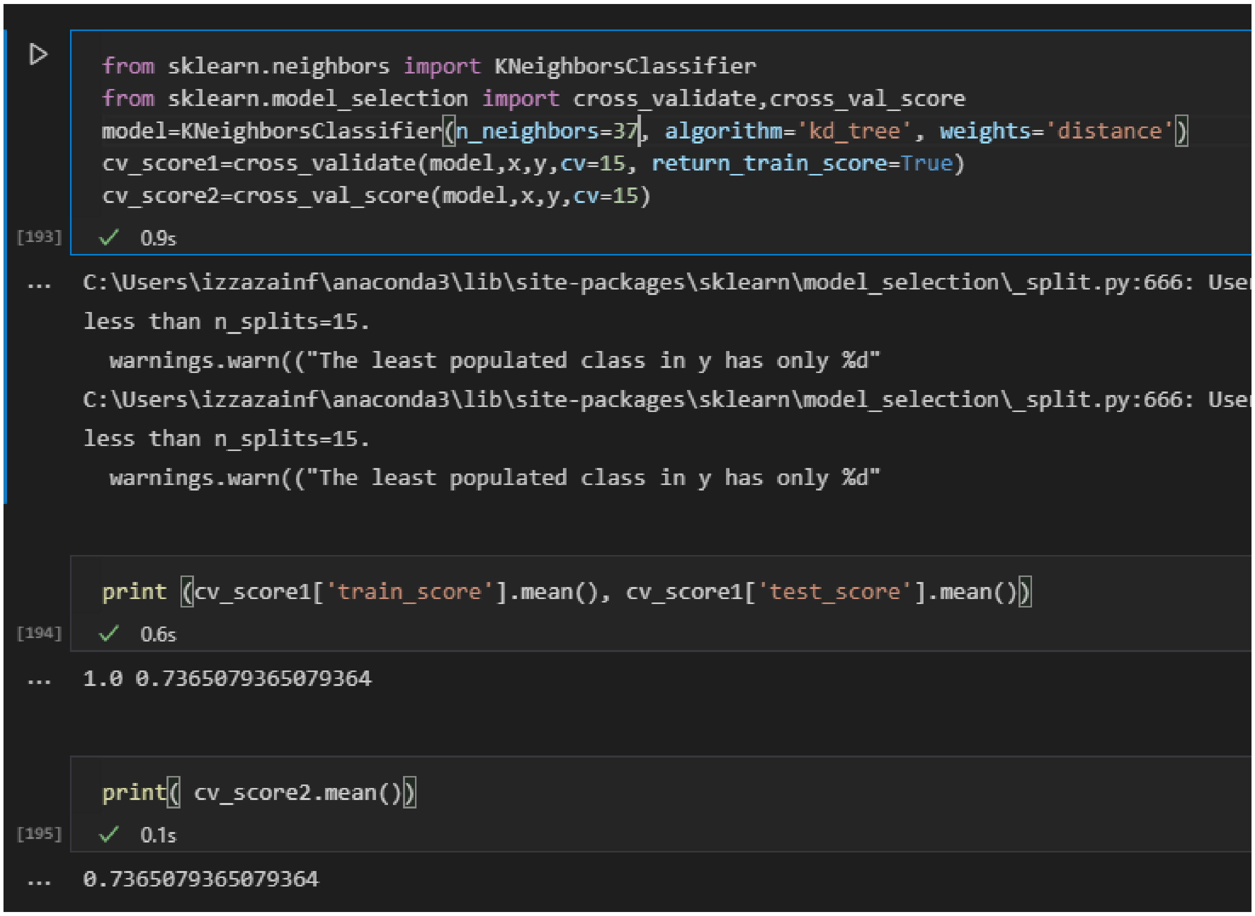
```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: U
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: U
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

```
print (cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
1.0 0.7031746031746031
```

```
print( cv_score2.mean())
```

```
0.7031746031746031
```



```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_validate,cross_val_score
model=KNeighborsClassifier(n_neighbors=37, algorithm='kd_tree', weights='distance')
cv_score1=cross_validate(model,x,y,cv=15, return_train_score=True)
cv_score2=cross_val_score(model,x,y,cv=15)
```

```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: Use
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: Use
less than n_splits=15.
  warnings.warn(("The least populated class in y has only %d"
```

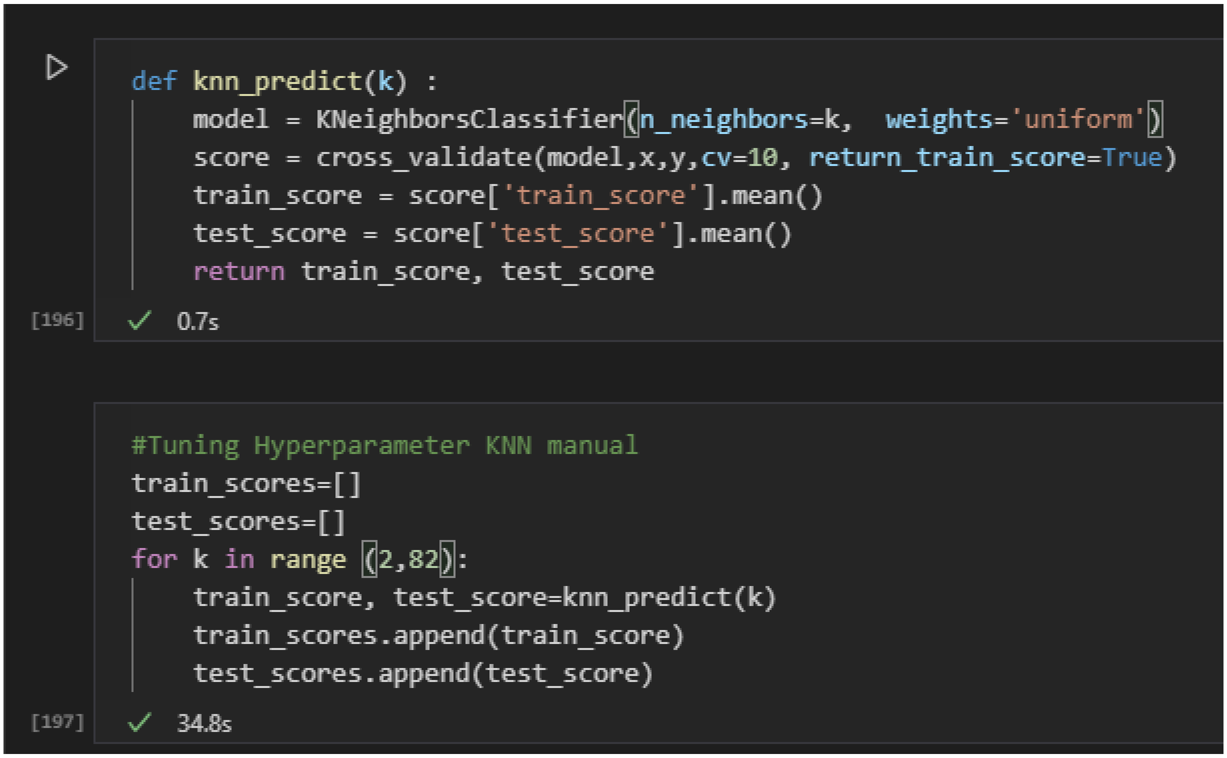
```
print (cv_score1['train_score'].mean(), cv_score1['test_score'].mean())
```

```
1.0 0.7365079365079364
```

```
print( cv_score2.mean())
```

```
0.7365079365079364
```

2. Manual dengan looping
   a. Weights: Uniform

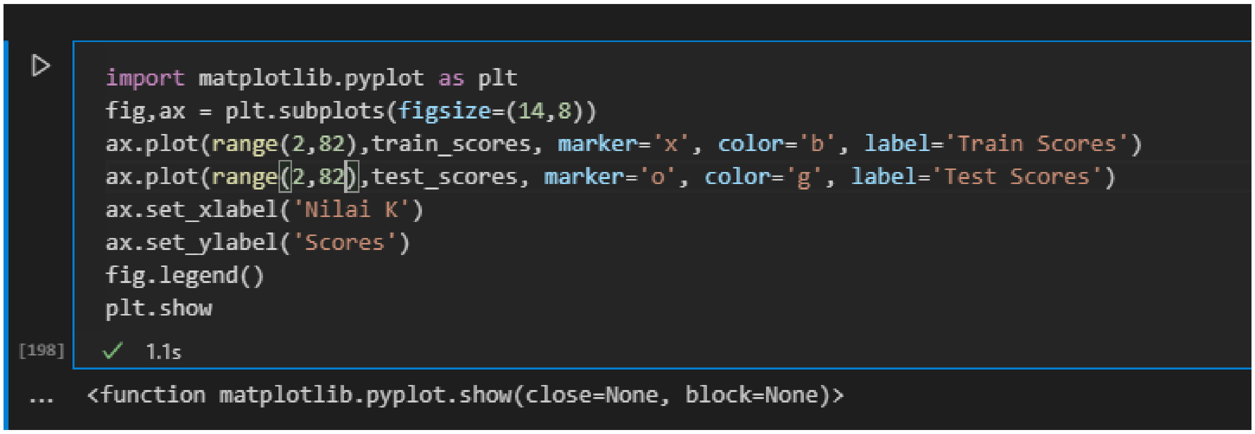


```
def knn_predict(k) :
    model = KNeighborsClassifier(n_neighbors=k,  weights='uniform')
    score = cross_validate(model,x,y,cv=10, return_train_score=True)
    train_score = score['train_score'].mean()
    test_score = score['test_score'].mean()
    return train_score, test_score
```

[196] ✓ 0.7s

```
#Tuning Hyperparameter KNN manual
train_scores=[]
test_scores=[]
for k in range (2,82):
    train_score, test_score=knn_predict(k)
    train_scores.append(train_score)
    test_scores.append(test_score)
```

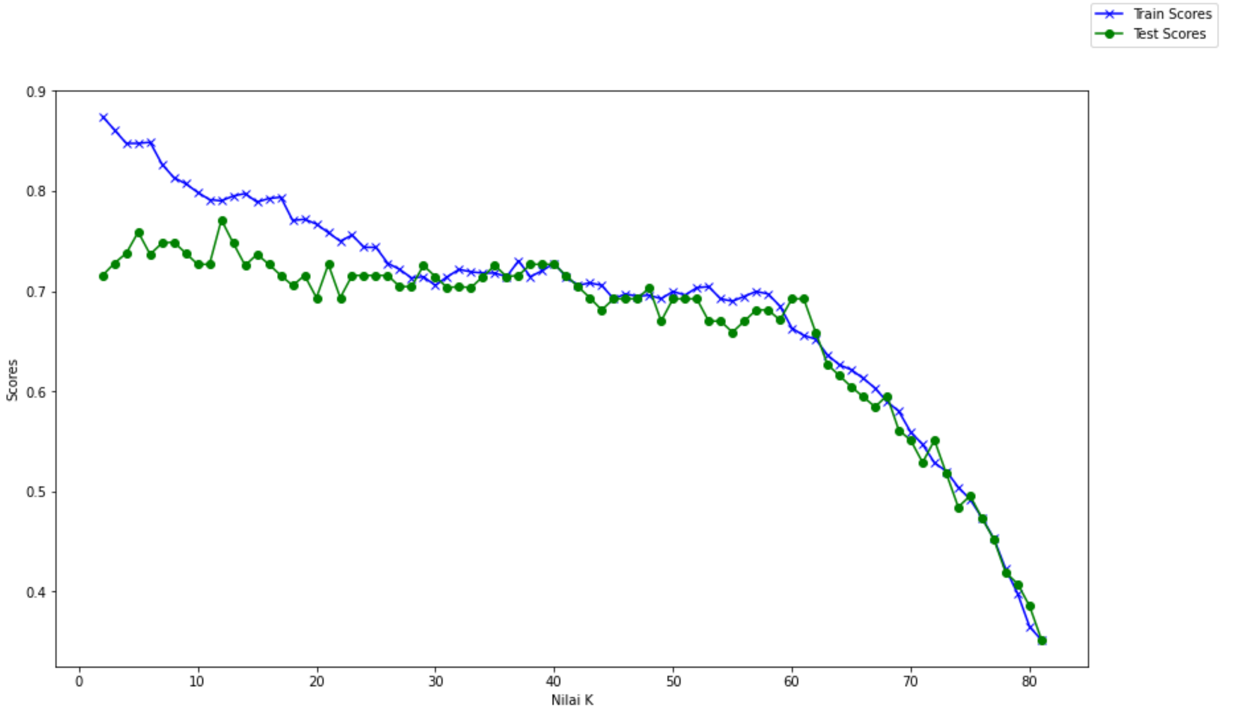
[197] ✓ 34.8s



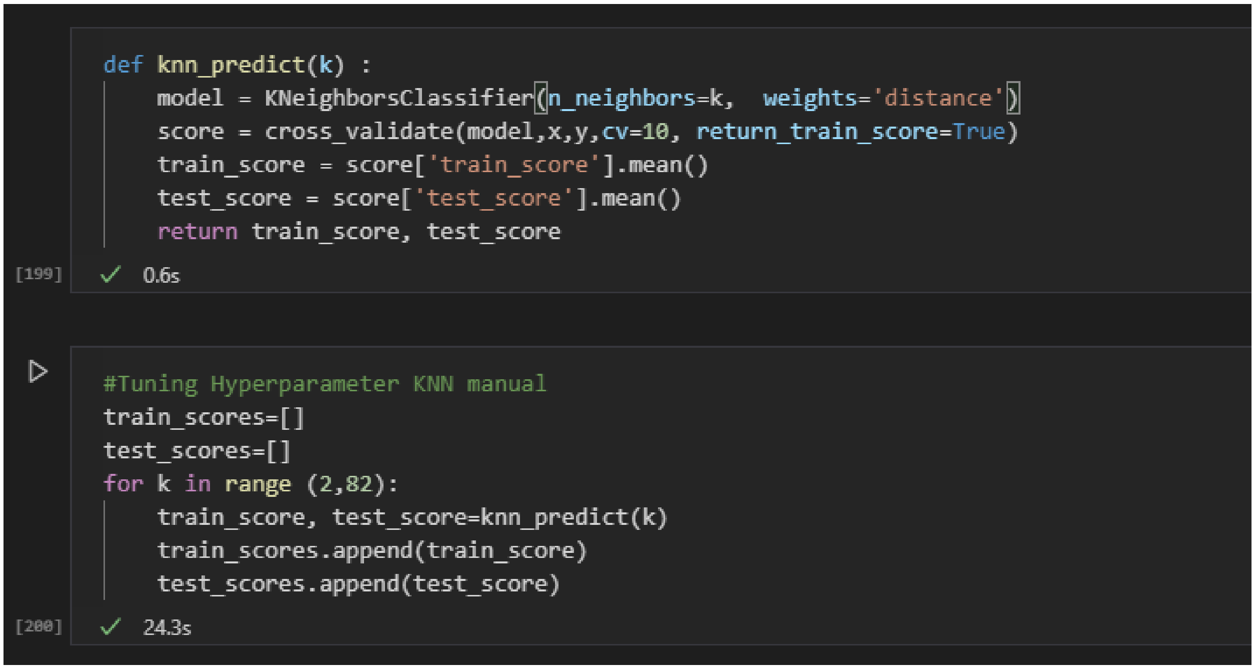
```
import matplotlib.pyplot as plt
fig,ax = plt.subplots(figsize=(14,8))
ax.plot(range(2,82),train_scores, marker='x', color='b', label='Train Scores')
ax.plot(range(2,82),test_scores, marker='o', color='g', label='Test Scores')
ax.set_xlabel('Nilai K')
ax.set_ylabel('Scores')
fig.legend()
plt.show
```

[198] ✓ 1.1s

```
... <function matplotlib.pyplot.show(close=None, block=None)>
```

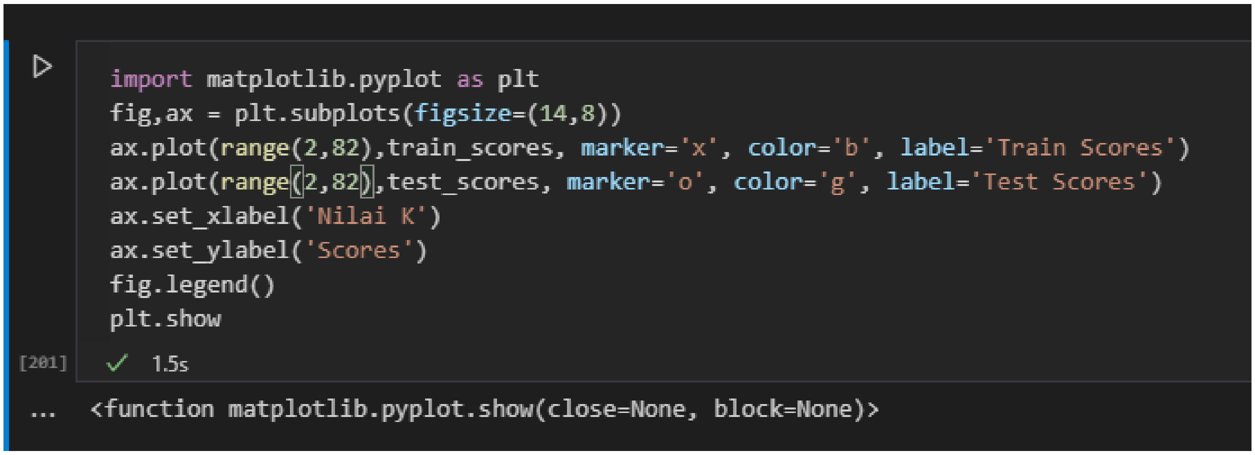


b. Weights: Distance



```
def knn_predict(k) :
    model = KNeighborsClassifier(n_neighbors=k,  weights='distance')
    score = cross_validate(model,x,y,cv=10, return_train_score=True)
    train_score = score['train_score'].mean()
    test_score = score['test_score'].mean()
    return train_score, test_score
[199]  ✓ 0.6s
```

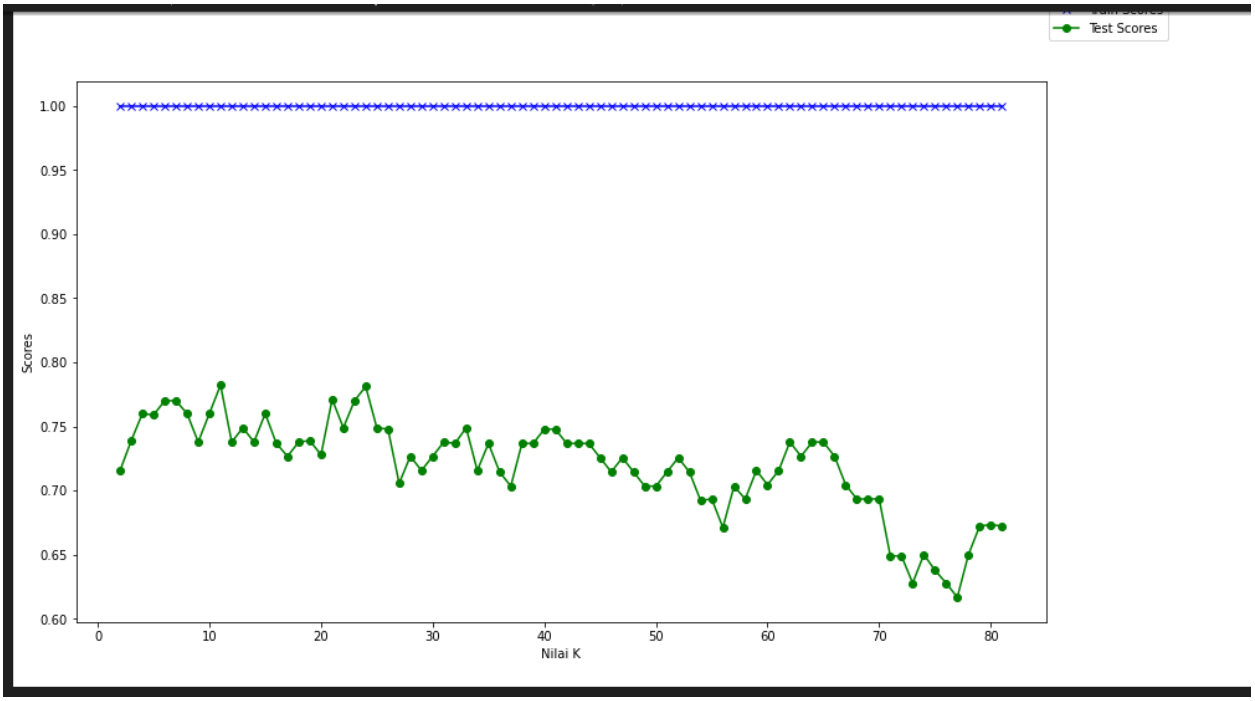
```
#Tuning Hyperparameter KNN manual
train_scores=[]
test_scores=[]
for k in range (2,82):
    train_score, test_score=knn_predict(k)
    train_scores.append(train_score)
    test_scores.append(test_score)
[200]  ✓ 24.3s
```



```
import matplotlib.pyplot as plt
fig,ax = plt.subplots(figsize=(14,8))
ax.plot(range(2,82),train_scores, marker='x', color='b', label='Train Scores')
ax.plot(range(2,82),test_scores, marker='o', color='g', label='Test Scores')
ax.set_xlabel('Nilai K')
ax.set_ylabel('Scores')
fig.legend()
plt.show
```

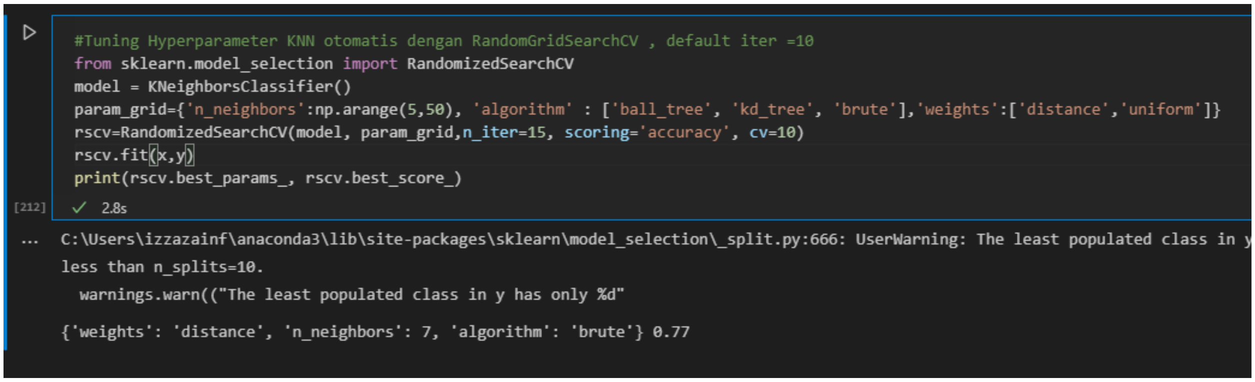
```
[201]  ✓ 1.5s
```

```
... <function matplotlib.pyplot.show(close=None, block=None)>
```



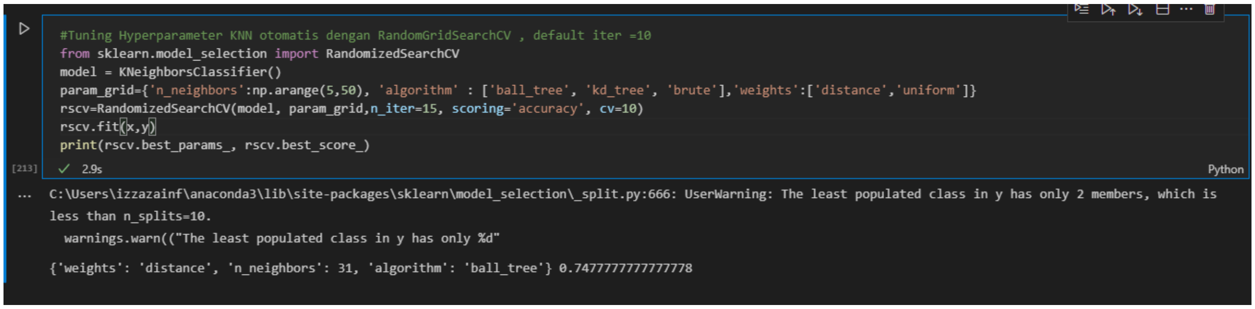
Metode Evaluasi Model : evaluasi model dilakukan dengan 2 metode

1. RandomGridSearchCV



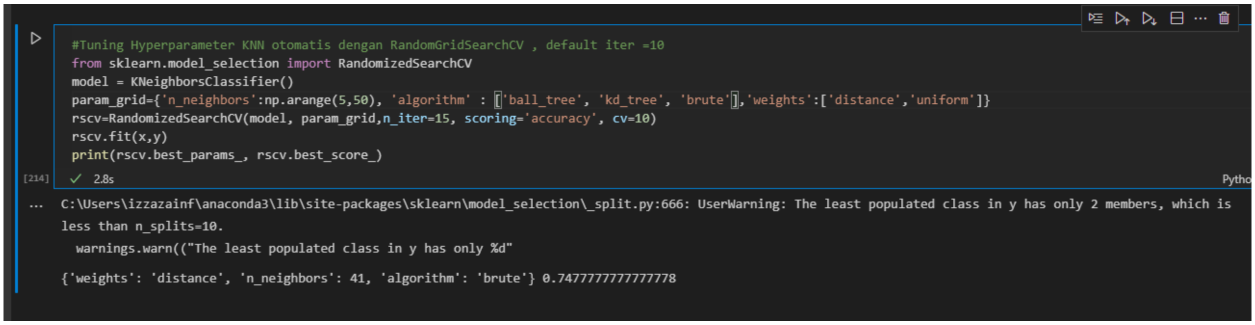
```
#Tuning Hyperparameter KNN otomatis dengan RandomGridSearchCV , default iter =10
from sklearn.model_selection import RandomizedSearchCV
model = KNeighborsClassifier()
param_grid={'n_neighbors':np.arange(5,50), 'algorithm' : ['ball_tree', 'kd_tree', 'brute'],'weights':['distance','uniform']}
rscv=RandomizedSearchCV(model, param_grid,n_iter=15, scoring='accuracy', cv=10)
rscv.fit(x,y)
print(rscv.best_params_, rscv.best_score_)
```

```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarning: The least populated class in y
less than n_splits=10.
  warnings.warn(("The least populated class in y has only %d"

{'weights': 'distance', 'n_neighbors': 7, 'algorithm': 'brute'} 0.77
```



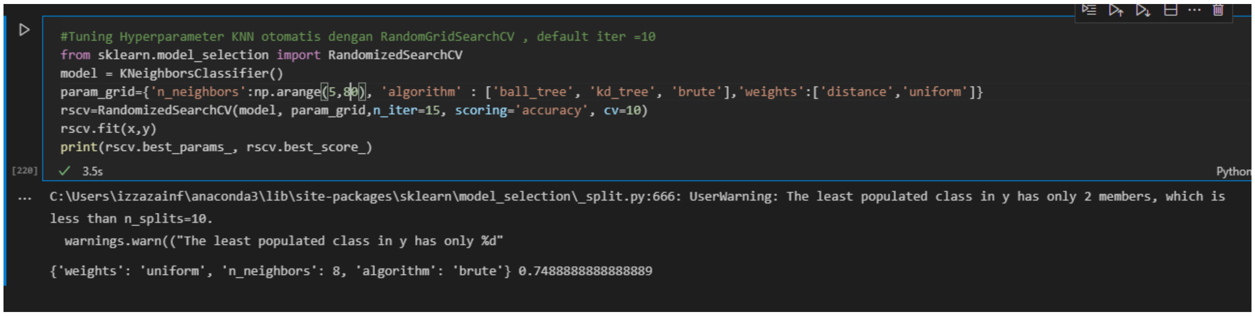
```
#Tuning Hyperparameter KNN otomatis dengan RandomGridSearchCV , default iter =10
from sklearn.model_selection import RandomizedSearchCV
model = KNeighborsClassifier()
param_grid={'n_neighbors':np.arange(5,50), 'algorithm' : ['ball_tree', 'kd_tree', 'brute'],'weights':['distance','uniform']}
rscv=RandomizedSearchCV(model, param_grid,n_iter=15, scoring='accuracy', cv=10)
rscv.fit(x,y)
print(rscv.best_params_, rscv.best_score_)
```

```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarning: The least populated class in y has only 2 members, which is
less than n_splits=10.
  warnings.warn(("The least populated class in y has only %d"

{'weights': 'distance', 'n_neighbors': 31, 'algorithm': 'ball_tree'} 0.7477777777777778
```



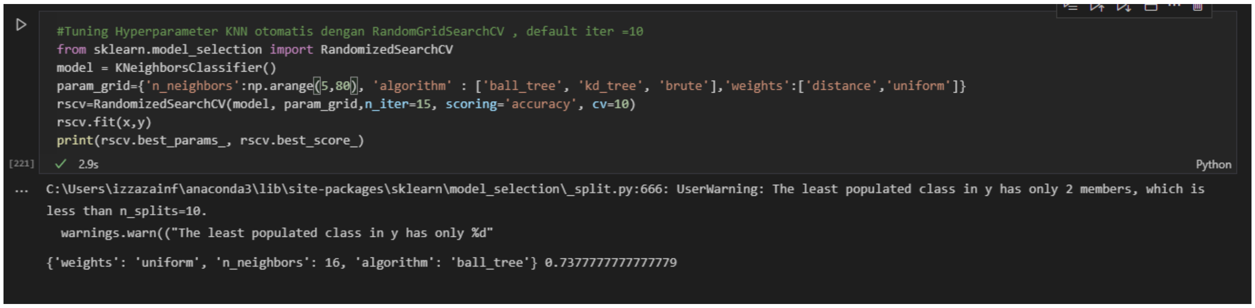
```
#Tuning Hyperparameter KNN otomatis dengan RandomGridSearchCV , default iter =10
from sklearn.model_selection import RandomizedSearchCV
model = KNeighborsClassifier()
param_grid={'n_neighbors':np.arange(5,50), 'algorithm' : ['ball_tree', 'kd_tree', 'brute'],'weights':['distance','uniform']}
rscv=RandomizedSearchCV(model, param_grid,n_iter=15, scoring='accuracy', cv=10)
rscv.fit(x,y)
print(rscv.best_params_, rscv.best_score_)
```

```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarning: The least populated class in y has only 2 members, which is
less than n_splits=10.
  warnings.warn(("The least populated class in y has only %d"

{'weights': 'distance', 'n_neighbors': 41, 'algorithm': 'brute'} 0.7477777777777778
```



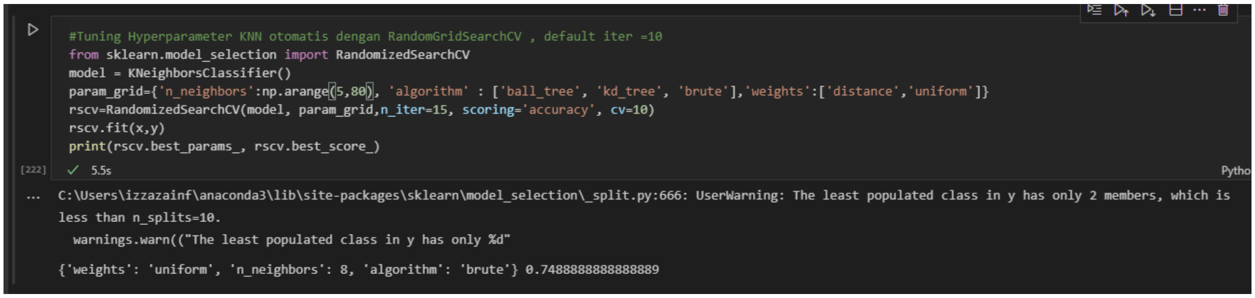
```
#Tuning Hyperparameter KNN otomatis dengan RandomGridSearchCV , default iter =10
from sklearn.model_selection import RandomizedSearchCV
model = KNeighborsClassifier()
param_grid={'n_neighbors':np.arange(5,80), 'algorithm' : ['ball_tree', 'kd_tree', 'brute'],'weights':['distance','uniform']}
rscv=RandomizedSearchCV(model, param_grid,n_iter=15, scoring='accuracy', cv=10)
rscv.fit(x,y)
print(rscv.best_params_, rscv.best_score_)
```

```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarning: The least populated class in y has only 2 members, which is
less than n_splits=10.
  warnings.warn(("The least populated class in y has only %d"

{'weights': 'uniform', 'n_neighbors': 8, 'algorithm': 'brute'} 0.7488888888888889
```



```
#Tuning Hyperparameter KNN otomatis dengan RandomGridSearchCV , default iter =10
from sklearn.model_selection import RandomizedSearchCV
model = KNeighborsClassifier()
param_grid={'n_neighbors':np.arange(5,80), 'algorithm' : ['ball_tree', 'kd_tree', 'brute'],'weights':['distance','uniform']}
rscv=RandomizedSearchCV(model, param_grid,n_iter=15, scoring='accuracy', cv=10)
rscv.fit(x,y)
print(rscv.best_params_, rscv.best_score_)
```

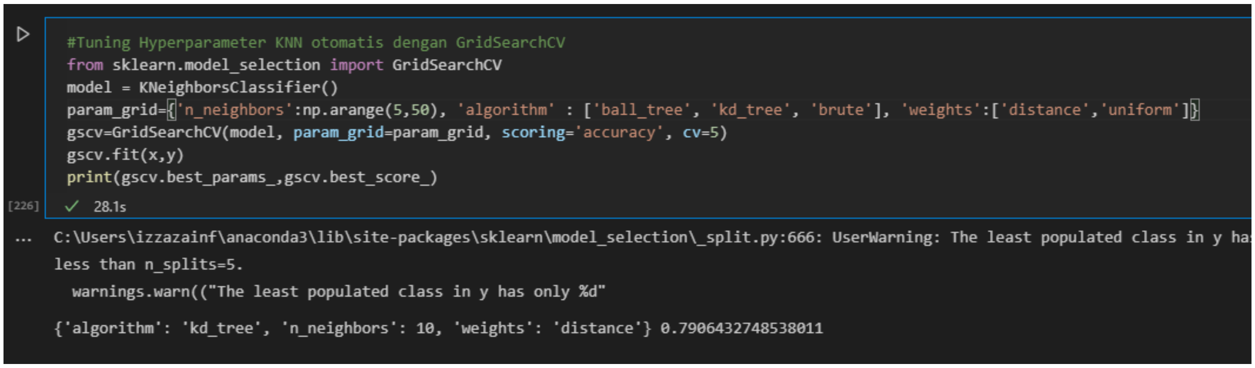
```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarning: The least populated class in y has only 2 members, which is
less than n_splits=10.
  warnings.warn(("The least populated class in y has only %d"

{'weights': 'uniform', 'n_neighbors': 16, 'algorithm': 'ball_tree'} 0.7377777777777779
```



```
#Tuning Hyperparameter KNN otomatis dengan RandomGridSearchCV , default iter =10
from sklearn.model_selection import RandomizedSearchCV
model = KNeighborsClassifier()
param_grid={'n_neighbors':np.arange(5,80), 'algorithm' : ['ball_tree', 'kd_tree', 'brute'],'weights':['distance','uniform']}
rscv=RandomizedSearchCV(model, param_grid,n_iter=15, scoring='accuracy', cv=10)
rscv.fit(x,y)
print(rscv.best_params_, rscv.best_score_)
```

```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarning: The least populated class in y has only 2 members, which is
less than n_splits=10.
  warnings.warn(("The least populated class in y has only %d"

{'weights': 'uniform', 'n_neighbors': 8, 'algorithm': 'brute'} 0.7488888888888889
```

2. GridSearchCV



```
#Tuning Hyperparameter KNN otomatis dengan GridSearchCV
from sklearn.model_selection import GridSearchCV
model = KNeighborsClassifier()
param_grid={'n_neighbors':np.arange(5,50), 'algorithm' : ['ball_tree', 'kd_tree', 'brute'], 'weights':['distance','uniform']}
gscv=GridSearchCV(model, param_grid=param_grid, scoring='accuracy', cv=5)
gscv.fit(x,y)
print(gscv.best_params_,gscv.best_score_)
```

```
C:\Users\izzazainf\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarning: The least populated class in y ha
less than n_splits=5.
  warnings.warn(("The least populated class in y has only %d"

{'algorithm': 'kd_tree', 'n_neighbors': 10, 'weights': 'distance'} 0.7906432748538011
```