In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
from sklearn.datasets import load_digits
digits=load_digits()
```

In [3]:
```
digits
```

Out[3]:
```
{'data': array([[ 0.,  0.,  5., ...,  0.,  0.,  0.],
        [ 0.,  0.,  0., ..., 10.,  0.,  0.],
        [ 0.,  0.,  0., ..., 16.,  9.,  0.],
        ...,
        [ 0.,  0.,  1., ...,  6.,  0.,  0.],
        [ 0.,  0.,  2., ..., 12.,  0.,  0.],
        [ 0.,  0., 10., ..., 12.,  1.,  0.]]),
 'target': array([0, 1, 2, ..., 8, 9, 8]),
 'frame': None,
 'feature_names': ['pixel_0_0',
  'pixel_0_1',
  'pixel_0_2',
  'pixel_0_3',
  'pixel_0_4',
  'pixel_0_5',
  'pixel_0_6',
  'pixel_0_7',
  'pixel_1_0',
  'pixel_1_1',
  'pixel_1_2',
  'pixel_1_3',
  'pixel_1_4',
  'pixel_1_5',
  'pixel_1_6',
  'pixel_1_7',
  'pixel_2_0',
  'pixel_2_1',
  'pixel_2_2',
  'pixel_2_3',
  'pixel_2_4',
  'pixel_2_5',
  'pixel_2_6',
  'pixel_2_7',
  'pixel_3_0',
  'pixel_3_1',
  'pixel_3_2',
  'pixel_3_3',
  'pixel_3_4',
  'pixel_3_5',
  'pixel_3_6',
  'pixel_3_7',
  'pixel_4_0',
  'pixel_4_1',
  'pixel_4_2',
  'pixel_4_3',
  'pixel_4_4',
  'pixel_4_5',
  'pixel_4_6',
  'pixel_4_7',
  'pixel_5_0',
  'pixel_5_1',
  'pixel_5_2',
  'pixel_5_3',
  'pixel_5_4',
  'pixel_5_5',
  'pixel_5_6',
  'pixel_5_7',
  'pixel_6_0',
  'pixel_6_1',
  'pixel_6_2',
  'pixel_6_3',
  'pixel_6_4',
  'pixel_6_5',
  'pixel_6_6',
```

```
 'pixel_6_7',
 'pixel_7_0',
 'pixel_7_1',
 'pixel_7_2',
 'pixel_7_3',
 'pixel_7_4',
 'pixel_7_5',
 'pixel_7_6',
 'pixel_7_7'],
'target_names': array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
'images': array([[[ 0.,  0.,  5., ...,  1.,  0.,  0.],
        [ 0.,  0., 13., ..., 15.,  5.,  0.],
        [ 0.,  3., 15., ..., 11.,  8.,  0.],
        ...,
        [ 0.,  4., 11., ..., 12.,  7.,  0.],
        [ 0.,  2., 14., ..., 12.,  0.,  0.],
        [ 0.,  0.,  6., ...,  0.,  0.,  0.]],

       [[ 0.,  0.,  0., ...,  5.,  0.,  0.],
        [ 0.,  0.,  0., ...,  9.,  0.,  0.],
        [ 0.,  0.,  3., ...,  6.,  0.,  0.],
        ...,
        [ 0.,  0.,  1., ...,  6.,  0.,  0.],
        [ 0.,  0.,  1., ...,  6.,  0.,  0.],
        [ 0.,  0.,  0., ..., 10.,  0.,  0.]],

       [[ 0.,  0.,  0., ..., 12.,  0.,  0.],
        [ 0.,  0.,  3., ..., 14.,  0.,  0.],
        [ 0.,  0.,  8., ..., 16.,  0.,  0.],
        ...,
        [ 0.,  9., 16., ...,  0.,  0.,  0.],
        [ 0.,  3., 13., ..., 11.,  5.,  0.],
        [ 0.,  0.,  0., ..., 16.,  9.,  0.]],

       ...,

       [[ 0.,  0.,  1., ...,  1.,  0.,  0.],
        [ 0.,  0., 13., ...,  2.,  1.,  0.],
        [ 0.,  0., 16., ..., 16.,  5.,  0.],
        ...,
        [ 0.,  0., 16., ..., 15.,  0.,  0.],
        [ 0.,  0., 15., ..., 16.,  0.,  0.],
        [ 0.,  0.,  2., ...,  6.,  0.,  0.]],

       [[ 0.,  0.,  2., ...,  0.,  0.,  0.],
        [ 0.,  0., 14., ..., 15.,  1.,  0.],
        [ 0.,  4., 16., ..., 16.,  7.,  0.],
        ...,
        [ 0.,  0.,  0., ..., 16.,  2.,  0.],
        [ 0.,  0.,  4., ..., 16.,  2.,  0.],
        [ 0.,  0.,  5., ..., 12.,  0.,  0.]],

       [[ 0.,  0., 10., ...,  1.,  0.,  0.],
        [ 0.,  2., 16., ...,  1.,  0.,  0.],
        [ 0.,  0., 15., ..., 15.,  0.,  0.],
        ...,
        [ 0.,  4., 16., ..., 16.,  6.,  0.],
        [ 0.,  8., 16., ..., 16.,  8.,  0.],
        [ 0.,  1.,  8., ..., 12.,  1.,  0.]]]),
 'DESCR': ".. _digits_dataset:\n\nOptical recognition of handwritten digits datase
t\n--------------------------------------------------\n\n**Data Set Characteristic
s:**\n\n    :Number of Instances: 1797\n    :Number of Attributes: 64\n    :Attrib
ute Information: 8x8 image of integer pixels in the range 0..16.\n    :Missing Att
ribute Values: None\n    :Creator: E. Alpaydin (alpaydin '@' boun.edu.tr)\n    :Da
```

```
te: July; 1998\n\nThis is a copy of the test set of the UCI ML hand-written digits
datasets\nhttps://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritt
en+Digits\n\nThe data set contains images of hand-written digits: 10 classes where
\neach class refers to a digit.\n\nPreprocessing programs made available by NIST w
ere used to extract\nnormalized bitmaps of handwritten digits from a preprinted fo
rm. From a\ntotal of 43 people, 30 contributed to the training set and different 1
3\nto the test set. 32x32 bitmaps are divided into nonoverlapping blocks of\n4x4 a
nd the number of on pixels are counted in each block. This generates\nan input mat
rix of 8x8 where each element is an integer in the range\n0..16. This reduces dime
nsionality and gives invariance to small\ndistortions.\n\nFor info on NIST preproc
essing routines, see M. D. Garris, J. L. Blue, G.\nT. Candela, D. L. Dimmick, J. G
eist, P. J. Grother, S. A. Janet, and C.\nL. Wilson, NIST Form-Based Handprint Rec
ognition System, NISTIR 5469,\n1994.\n\n.. topic:: References\n\n  - C. Kaynak (19
95) Methods of Combining Multiple Classifiers and Their\n    Applications to Handw
ritten Digit Recognition, MSc Thesis, Institute of\n    Graduate Studies in Scienc
e and Engineering, Bogazici University.\n  - E. Alpaydin, C. Kaynak (1998) Cascadi
ng Classifiers, Kybernetika.\n  - Ken Tang and Ponnuthurai N. Suganthan and Xi Yao
and A. Kai Qin.\n    Linear dimensionalityreduction using relevance weighted LDA.
School of\n    Electrical and Electronic Engineering Nanyang Technological Univers
ity.\n    2005.\n  - Claudio Gentile. A New Approximate Maximal Margin Classificat
ion\n    Algorithm. NIPS. 2000.\n"}
```

In [6]:
```
df=pd.DataFrame(digits.data,columns=digits.feature_names)
```

In [7]:
```
df.head()
```

Out[7]:

	pixel_0_0	pixel_0_1	pixel_0_2	pixel_0_3	pixel_0_4	pixel_0_5	pixel_0_6	pixel_0_7	pixel_1_0	p
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0	0.0	0.0	
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0	0.0	0.0	
2	0.0	0.0	0.0	4.0	15.0	12.0	0.0	0.0	0.0	
3	0.0	0.0	7.0	15.0	13.0	1.0	0.0	0.0	0.0	
4	0.0	0.0	0.0	1.0	11.0	0.0	0.0	0.0	0.0	

5 rows × 64 columns

In [9]:
```
df.shape
```

Out[9]:
```
(1797, 64)
```

In [10]:
```
digits.target
```

Out[10]:
```
array([0, 1, 2, ..., 8, 9, 8])
```

In [11]:
```
df['digits']=digits.target
```

In [12]:
```
df.head(2)
```

Out[12]:

	pixel_0_0	pixel_0_1	pixel_0_2	pixel_0_3	pixel_0_4	pixel_0_5	pixel_0_6	pixel_0_7	pixel_1_0	p
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0	0.0	0.0	
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0	0.0	0.0	

2 rows × 65 columns

In [16]:
```
df['digits'].value_counts()
```

Out[16]:
```
3    183
1    182
5    182
4    181
6    181
9    180
7    179
0    178
2    177
8    174
Name: digits, dtype: int64
```

In [17]:
```
df.isnull().sum().sum()
```

Out[17]:
```
0
```

In [18]:
```
x=df.drop(['digits'],axis=1)
y=df['digits']
```

In [20]:
```
from sklearn.model_selection import train_test_split
```

In [21]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=1)
```

In [24]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [28]:
```
rfc1=RandomForestClassifier(n_estimators=200,max_depth=2)
```

In [29]:
```
rfc1.fit(x_train,y_train)
```

Out[29]:
```
RandomForestClassifier(max_depth=2, n_estimators=200)
```

In [30]:
```
y_predict=rfc1.predict(x_test)
```

In [33]:
```
from sklearn.metrics import confusion_matrix,accuracy_score
```

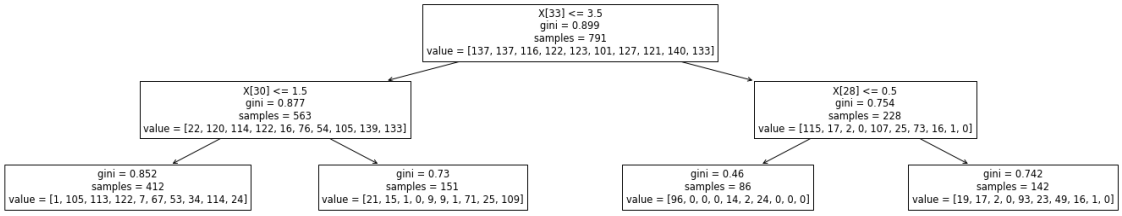
In [34]:
```
confusion_matrix(y_test,y_predict)
accuracy_score(y_test,y_predict)
rfc1.estimators_[0]
```

Out[34]:
```
DecisionTreeClassifier(max_depth=2, max_features='auto', random_state=786542976)
```

In [35]:
```
from sklearn import tree
```

In [36]:
```
import matplotlib.pyplot as plt
```

In [41]:
```
plt.figure(figsize=(25,5))
tree.plot_tree(rfc1.estimators_[0])
plt.show()
```



In [ ]: