mk 02-09-2023

In [45]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [46]:
```
from sklearn.linear_model import LogisticRegression
a=pd.read_csv(r"C:\USERS\user\Downloads\C8_loan-test.csv")
a
```

1	LP001022	Male	Yes	1	Graduate	No	3076	15
2	LP001031	Male	Yes	2	Graduate	No	5000	18
3	LP001035	Male	Yes	2	Graduate	No	2340	25
4	LP001051	Male	No	0	Not Graduate	No	3276	
...	...	...	...	...	...	...	...	
362	LP002971	Male	Yes	3+	Not Graduate	Yes	4009	17
363	LP002975	Male	Yes	0	Graduate	No	4158	7
364	LP002980	Male	No	0	Graduate	No	3250	19
365	LP002986	Male	Yes	0	Graduate	No	5000	23
366	LP002989	Male	No	0	Graduate	Yes	9200	

367 rows × 12 columns

In [47]:
```
a=a.head(10)
a
```

Out[47]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome
0	LP001015	Male	Yes	0	Graduate	No	5720	0
1	LP001022	Male	Yes	1	Graduate	No	3076	1500
2	LP001031	Male	Yes	2	Graduate	No	5000	1800
3	LP001035	Male	Yes	2	Graduate	No	2340	2546
4	LP001051	Male	No	0	Not Graduate	No	3276	0
5	LP001054	Male	Yes	0	Not Graduate	Yes	2165	3422
6	LP001055	Female	No	1	Not Graduate	No	2226	0
7	LP001056	Male	Yes	2	Not Graduate	No	3881	0
8	LP001059	Male	Yes	2	Graduate	NaN	13633	0
9	LP001067	Male	No	0	Not Graduate	No	2400	2400

In [48]:
```
from sklearn.linear_model import LogisticRegression
```

In [49]:
```
a.columns
```

Out[49]:
```
Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
       'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
       'Loan_Amount_Term', 'Credit_History', 'Property_Area'],
      dtype='object')
```

In [50]:
```
b=a[['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
        'Loan_Amount_Term']]
b
```

Out[50]:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term
0	5720	0	110.0	360.0
1	3076	1500	126.0	360.0
2	5000	1800	208.0	360.0
3	2340	2546	100.0	360.0
4	3276	0	78.0	360.0
5	2165	3422	152.0	360.0
6	2226	0	59.0	360.0
7	3881	0	147.0	360.0
8	13633	0	280.0	240.0
9	2400	2400	123.0	360.0

In [51]:
```
c=b.iloc[:,0:15]
d=b.iloc[:,-1]
```

In [52]:
```
c.shape
```

Out[52]:
```
(10, 4)
```

In [53]:
```
d.shape
```

Out[53]:
```
(10,)
```

In [54]:
```
from sklearn.preprocessing import StandardScaler
fs=StandardScaler().fit_transform(c)
fs
```

Out[54]:
```
array([[ 0.40915196, -0.92743548, -0.46043293,  0.33333333],
       [-0.39319008,  0.26484531, -0.20011749,  0.33333333],
       [ 0.19066244,  0.50330146,  1.13399913,  0.33333333],
       [-0.61653492,  1.09626244, -0.62313008,  0.33333333],
       [-0.33249855, -0.92743548, -0.98106381,  0.33333333],
       [-0.66964001,  1.79255442,  0.22289509,  0.33333333],
       [-0.65112909, -0.92743548, -1.2901884 ,  0.33333333],
       [-0.14890667, -0.92743548,  0.14154652,  0.33333333],
       [ 2.81041237, -0.92743548,  2.30541861, -3.        ],
       [-0.59832746,  0.98021378, -0.24892664,  0.33333333]])
```

In [55]:

```
logr=LogisticRegression()
logr.fit(fs,d)
```

Out[55]:

```
LogisticRegression()
```

In [56]:

```
e=[[2,5,77,8]]
```

In [57]:

```
prediction=logr.predict(e)
prediction
```

Out[57]:

```
array([240.])
```

In [58]:

```
logr.classes_
```

Out[58]:

```
array([240., 360.])
```

In [59]:

```
logr.predict_proba(e)[0][0]
```

Out[59]:

```
0.9999999999973238
```

In [60]:

```
import re
from sklearn.datasets import load_digits
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [61]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
```

In [62]:

```
digits=load_digits()
digits
```

Out[62]:

```
{'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',
```

In [63]:

```
plt.figure(figsize=(20,4))
```

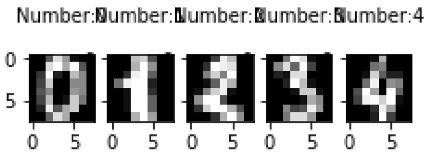
Out[63]:

```
<Figure size 1440x288 with 0 Axes>

<Figure size 1440x288 with 0 Axes>
```

In [64]:

```
for index,(image,label) in enumerate(zip(digits.data[0:5],digits.target[0:5])):
    plt.subplot(1,8,index+1)
    plt.imshow(np.reshape(image,(8
                                 ,8)),cmap=plt.cm.gray)
    plt.title('Number:%i\n'%label,fontsize=10)
```



In [65]:

```
x_train,x_test,y_train,y_test=train_test_split(digits.data,digits.target,test_size=0
```

In [66]:

```
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(1257, 64)
(540, 64)
(1257,)
(540,)
```

In [67]:

```
logre=LogisticRegression(max_iter=10000)
logre.fit(x_train,y_train)

```

Out[67]: `LogisticRegression(max_iter=10000)`

In [68]:

```
print(logre.predict(x_test))
```

```
[4 4 5 1 5 5 2 7 2 1 7 1 5 2 0 5 9 0 5 7 4 3 9 0 4 7 9 0 0 7 8 6 7 0 0 0 5
 3 2 7 3 4 4 0 0 4 0 2 5 2 2 3 6 3 2 0 6 4 7 0 1 8 6 7 2 9 9 5 0 7 2 6 1 7
 7 6 1 2 6 8 7 2 2 8 4 0 1 0 2 2 3 7 6 0 9 1 9 4 6 8 6 9 4 1 6 8 2 8 4 5 9
 7 3 4 5 5 0 3 4 5 0 7 5 8 1 7 3 8 1 6 8 4 4 6 4 6 1 9 9 9 2 6 6 0 9 3 0 1
 4 7 5 1 0 5 1 9 7 2 0 1 3 7 9 7 2 0 7 7 4 4 1 8 8 5 7 7 2 3 7 5 9 2 2 8 4
 9 4 9 1 5 2 2 6 0 5 3 2 2 4 3 1 6 0 3 9 9 9 3 3 4 3 9 5 0 7 8 7 9 1 8 9 4
 0 1 7 1 0 4 2 9 3 3 0 3 2 7 7 8 9 9 6 0 4 0 7 3 4 5 7 5 3 0 8 0 0 7 7 6 5
 3 2 8 4 8 6 9 4 0 9 1 0 2 3 5 8 5 1 2 7 4 7 2 4 6 5 9 6 0 2 4 6 3 3 0 3 3
 4 7 0 6 0 5 2 2 6 9 7 9 7 9 0 7 0 0 6 5 5 4 9 8 2 9 6 5 5 8 6 7 3 1 0 2 6
 5 6 9 0 5 3 6 0 9 1 4 6 1 4 8 6 0 2 9 1 4 0 5 2 4 7 5 2 9 7 1 2 2 7 0 2 2
 4 9 8 6 9 3 0 5 8 3 0 4 9 0 2 0 2 7 8 4 9 6 1 2 0 9 4 8 0 5 2 0 0 6 0 9 2
 6 8 3 8 9 0 8 8 6 8 6 4 8 8 4 2 2 8 5 7 5 1 3 6 3 8 5 4 7 9 3 5 8 5 6 6 1
 4 1 0 6 3 7 5 0 3 6 3 0 6 5 9 0 3 9 8 7 3 1 4 2 8 7 3 7 6 1 8 0 0 4 0 7 1
 1 3 2 2 5 1 5 4 1 5 3 0 6 1 7 3 1 8 1 4 7 6 4 9 1 9 4 3 4 1 1 9 0 7 3 8 5
 8 7 6 5 2 0 3 7 2 6 2 7 7 0 9 1 8 6 8 1 6 9]
```

In [69]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [70]:

```
a=pd.read_csv(r"C:\USERS\user\Downloads\C8_loan-test.csv")
```

In [71]:

```
a=a.head(10)
a
```

0	LP001015	Male	Yes	0	Graduate	No	5720	(
1	LP001022	Male	Yes	1	Graduate	No	3076	150(
2	LP001031	Male	Yes	2	Graduate	No	5000	180(
3	LP001035	Male	Yes	2	Graduate	No	2340	254(
4	LP001051	Male	No	0	Not Graduate	No	3276	(
5	LP001054	Male	Yes	0	Not Graduate	Yes	2165	342
6	LP001055	Female	No	1	Not Graduate	No	2226	(
7	LP001056	Male	Yes	2	Not Graduate	No	3881	(
8	LP001059	Male	Yes	2	Graduate	NaN	13633	(
9	LP001067	Male	No	0	Not Graduate	No	2400	240(

In [72]:

```
b=a[['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
       'Loan_Amount_Term',  'Property_Area']]
b
```

Out[72]:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Property_Area
0	5720	0	110.0	360.0	Urban
1	3076	1500	126.0	360.0	Urban
2	5000	1800	208.0	360.0	Urban
3	2340	2546	100.0	360.0	Urban
4	3276	0	78.0	360.0	Urban
5	2165	3422	152.0	360.0	Urban
6	2226	0	59.0	360.0	Semiurban
7	3881	0	147.0	360.0	Rural
8	13633	0	280.0	240.0	Urban
9	2400	2400	123.0	360.0	Semiurban

In [73]:

```
b['Property_Area'].value_counts()
```

Out[73]:

```
Urban        7
Semiurban    2
Rural        1
Name: Property_Area, dtype: int64
```

In [74]:

```
x=b.drop('Property_Area',axis=1)
y=b['Property_Area']
print(b)
```

```
   ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
0             5720                  0       110.0             360.0   
1             3076               1500       126.0             360.0   
2             5000               1800       208.0             360.0   
3             2340               2546       100.0             360.0   
4             3276                  0        78.0             360.0   
5             2165               3422       152.0             360.0   
6             2226                  0        59.0             360.0   
7             3881                  0       147.0             360.0   
8            13633                  0       280.0             240.0   
9             2400               2400       123.0             360.0   

  Property_Area  
0         Urban  
1         Urban  
2         Urban  
3         Urban  
4         Urban  
5         Urban  
6     Semiurban  
7         Rural  
8         Urban  
9     Semiurban  
```

In [75]:

```
g1={"Property_Area":{'Urban':1,'Semiurban':2,'Rural':3}}
a=a.replace(g1)
print(a)
```

```
    Loan_ID  Gender Married Dependents     Education Self_Employed  \
0  LP001015    Male     Yes          0      Graduate            No
1  LP001022    Male     Yes          1      Graduate            No
2  LP001031    Male     Yes          2      Graduate            No
3  LP001035    Male     Yes          2      Graduate            No
4  LP001051    Male      No          0  Not Graduate            No
5  LP001054    Male     Yes          0  Not Graduate           Yes
6  LP001055  Female      No          1  Not Graduate            No
7  LP001056    Male     Yes          2  Not Graduate            No
8  LP001059    Male     Yes          2      Graduate           NaN
9  LP001067    Male      No          0  Not Graduate            No

   ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
0             5720                  0       110.0             360.0
1             3076               1500       126.0             360.0
2             5000               1800       208.0             360.0
3             2340               2546       100.0             360.0
4             3276                  0        78.0             360.0
5             2165               3422       152.0             360.0
6             2226                  0        59.0             360.0
7             3881                  0       147.0             360.0
8            13633                  0       280.0             240.0
9             2400               2400       123.0             360.0

   Credit_History  Property_Area
0             1.0              1
1             1.0              1
2             1.0              1
3             NaN              1
4             1.0              1
5             1.0              1
6             1.0              2
7             0.0              3
8             1.0              1
9             1.0              2
```

In [76]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

In [77]:

```
from sklearn.ensemble import RandomForestClassifier
```

In [78]:

```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[78]:

```
RandomForestClassifier()
```

In [79]:

```
parameters={'max_depth':[1,2,3,4,5],
           'min_samples_leaf':[5,10,15,20,25],
           'n_estimators':[10,20,30,40,50]}
```

In [80]:

```
from sklearn.model_selection import GridSearchCV
```

In [81]:

```
grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy"
grid_search.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserW
arning: The least populated class in y has only 1 members, which is less than n_splits=
2.
  warnings.warn(("The least populated class in y has only %d"
```

Out[81]:

```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [1, 2, 3, 4, 5],
                         'min_samples_leaf': [5, 10, 15, 20, 25],
                         'n_estimators': [10, 20, 30, 40, 50]},
             scoring='accuracy')
```

In [82]:

```
grid_search.best_score_
```

Out[82]:

```
0.5833333333333333
```

In [83]:

```
rfc_best=grid_search.best_estimator_
```

In [84]:

```
from sklearn.tree import plot_tree
```

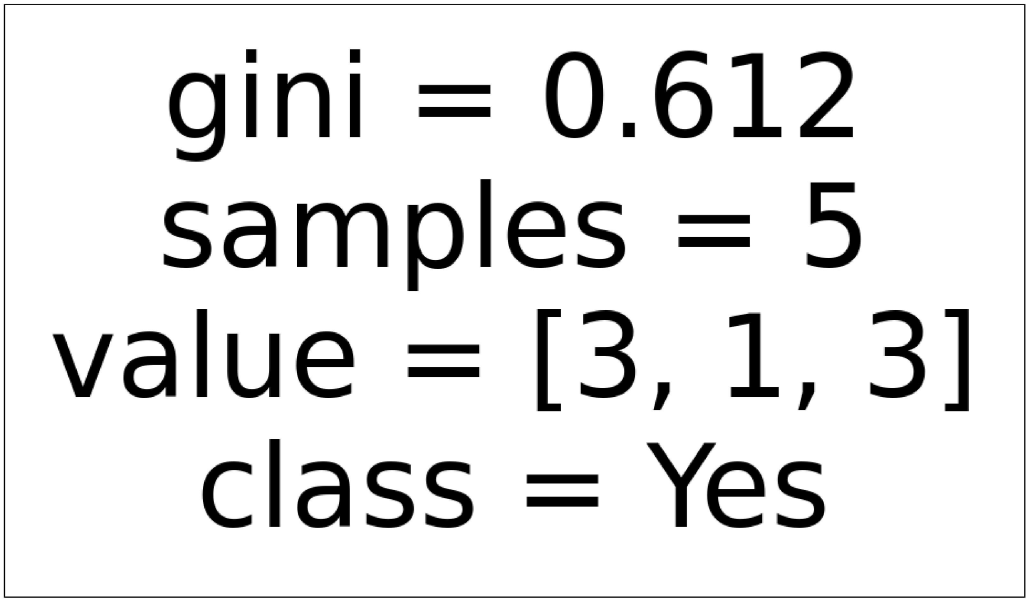
In [90]:

```
plt.figure(figsize=(20,10))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','No','1

```

Out[90]:

```
[Text(558.0, 271.8, 'gini = 0.612\nsamples = 5\nvalue = [3, 1, 3]\nclass = Yes')]
```



In [ ]:

```

```