mk 02-09-2023

In [342]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [382]:
```
from sklearn.linear_model import LogisticRegression
a=pd.read_csv(r"C:\USERS\user\Downloads\C8_loan-train.csv")
a
```

1	LP001003	Male	Yes	1	Graduate	No	4583	150
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	235
4	LP001008	Male	No	0	Graduate	No	6000	
...	...	...	...	...	...	...	...	
609	LP002978	Female	No	0	Graduate	No	2900	
610	LP002979	Male	Yes	3+	Graduate	No	4106	
611	LP002983	Male	Yes	1	Graduate	No	8072	24
612	LP002984	Male	Yes	2	Graduate	No	7583	
613	LP002990	Female	No	0	Graduate	Yes	4583	

614 rows × 13 columns

In [452]:
```
a=a.head(100)
a
```

Female	No	0	Graduate	No	4000	2275.0	144.0
Female	Yes	0	Not Graduate	No	1928	1644.0	100.0
Female	No	0	Graduate	No	3086	0.0	120.0
Female	No	0	Graduate	No	4230	0.0	112.0
Male	Yes	2	Graduate	No	4616	0.0	134.0
Female	Yes	1	Graduate	Yes	11500	0.0	286.0
Male	Yes	2	Graduate	No	2708	1167.0	97.0
Male	Yes	0	Graduate	No	2132	1591.0	96.0
Male	Yes	0	Graduate	No	3366	2200.0	135.0
Male	Yes	1	Graduate	No	8080	2250.0	180.0
Male	Yes	2	Not Graduate	No	3357	2859.0	144.0

In [453]:
```
from sklearn.linear_model import LogisticRegression
```

In [454]:
```
a.columns
```

Out[454]:
```
Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
       'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
       'Loan_Amount_Term', 'Credit_History', 'Property_Area', 'Loan_Status'],
      dtype='object')
```

In [455]:
```
b=a[['ApplicantIncome', 'CoapplicantIncome']]
b
```

Out[455]:

	ApplicantIncome	CoapplicantIncome
0	5849	0.0
1	4583	1508.0
2	3000	0.0
3	2583	2358.0
4	6000	0.0
5	5417	4196.0
6	2333	1516.0
7	3036	2504.0
8	4006	1526.0
9	12841	10968.0
10	3200	700.0
11	2500	1840.0
12	3073	8106.0
13	1853	2840.0
14	1299	1086.0
15	4950	0.0
16	3596	0.0
17	3510	0.0
18	4887	0.0
19	2600	3500.0
20	7660	0.0
21	5955	5625.0
22	2600	1911.0
23	3365	1917.0
24	3717	2925.0
25	9560	0.0
26	2799	2253.0
27	4226	1040.0
28	1442	0.0
29	3750	2083.0
30	4166	3369.0
31	3167	0.0
32	4692	0.0
33	3500	1667.0
34	12500	3000.0
35	2275	2067.0
36	1828	1330.0
37	3667	1459.0
38	4166	7210.0

	ApplicantIncome	CoapplicantIncome
39	3748	1668.0
40	3600	0.0
41	1800	1213.0
42	2400	0.0
43	3941	2336.0
44	4695	0.0
45	3410	0.0
46	5649	0.0
47	5821	0.0
48	2645	3440.0
49	4000	2275.0
50	1928	1644.0
51	3086	0.0
52	4230	0.0
53	4616	0.0
54	11500	0.0
55	2708	1167.0
56	2132	1591.0
57	3366	2200.0
58	8080	2250.0
59	3357	2859.0

In [456]:
```
c=b.iloc[:,0:5]
d=b.iloc[:,-1]
```

In [457]:
```
c.shape
```

Out[457]: (60, 2)

In [458]:
```
d.shape
```

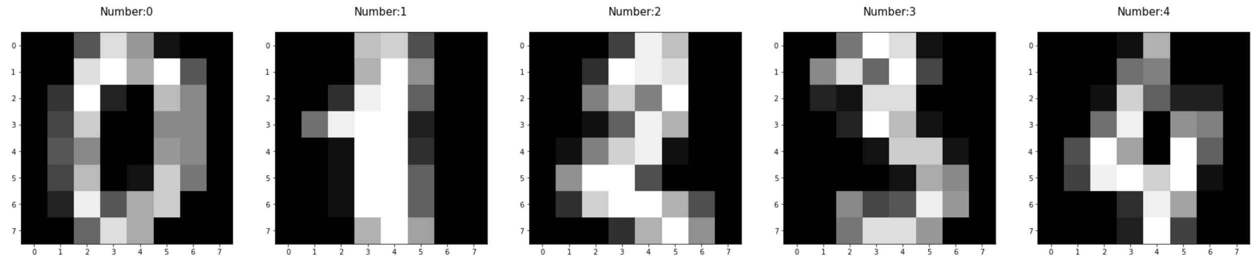
Out[458]: (60,)

In [459]:
```
from sklearn.preprocessing import StandardScaler
fs=StandardScaler().fit_transform(c)
fs
```

```
Out[459]: array([[ 0.67433211, -0.82097989],
           [ 0.15206627, -0.1008207 ],
           [-0.5009723 , -0.82097989],
           [-0.67299826,  0.30510458],
           [ 0.73662448, -0.82097989],
           [ 0.49611817,  1.18285829],
           [-0.77613132, -0.09700022],
           [-0.48612114,  0.37482822],
           [-0.08596485, -0.09222463],
           [ 3.55875768,  4.41688885],
           [-0.41846585, -0.48668849],
           [-0.70723843,  0.05772894],
           [-0.47085745,  3.05011456],
           [-0.97414681,  0.53528809],
           [-1.20268968, -0.30235066],
           [ 0.30346561, -0.82097989],
           [-0.25510307, -0.82097989],
           [-0.29058085, -0.82097989],
           [ 0.27747607, -0.82097989],
           [-0.66598521,  0.85047713],
           [ 1.42142804, -0.82097989],
           [ 0.71806053,  1.86529032],
           [-0.66598521,  0.09163564],
           [-0.35039803,  0.094501  ],
           [-0.20518667,  0.57588062],
           [ 2.20523933, -0.82097989],
           [-0.58389129,  0.25496087],
           [ 0.00479225, -0.32431838],
           [-1.14369757, -0.82097989],
           [-0.19157311,  0.17377581],
           [-0.01995969,  0.78791688],
           [-0.43207942, -0.82097989],
           [ 0.19703228, -0.82097989],
           [-0.29470617, -0.02488879],
           [ 3.41808418,  0.61169755],
           [-0.80005819,  0.16613487],
           [-0.98446011, -0.18582622],
           [-0.22581328, -0.12422109],
           [-0.01995969,  2.62222157],
           [-0.19239817, -0.02441123],
           [-0.25345295, -0.82097989],
           [-0.99601102, -0.24170064],
           [-0.74849166, -0.82097989],
           [-0.11277944,  0.29459828],
           [ 0.19826988, -0.82097989],
           [-0.33183408, -0.82097989],
           [ 0.59182566, -0.82097989],
           [ 0.66278121, -0.82097989],
           [-0.64742126,  0.82182358],
           [-0.08844004,  0.26546717],
           [-0.94320689, -0.03587265],
           [-0.46549453, -0.82097989],
           [ 0.00644238, -0.82097989],
           [ 0.16567983, -0.82097989],
           [ 3.00555192, -0.82097989],
           [-0.62143172, -0.26366836],
           [-0.85905031, -0.06118329],
           [-0.3499855 ,  0.22965023],
           [ 1.59469159,  0.25352819],
           [-0.35369829,  0.54436171]])
```

In [460]:
```
logr=LogisticRegression()
logr.fit(fs,d)
```
Out[460]:
```
LogisticRegression()
```
In [461]:
```
e=[[777,55]]
```
In [462]:
```
prediction=logr.predict(e)
prediction
```
Out[462]:
```
array([3000.])
```
In [463]:
```
logr.classes_
```
Out[463]:
```
array([    0.,   700.,  1040.,  1086.,  1167.,  1213.,  1330.,  1459.,
        1508.,  1516.,  1526.,  1591.,  1644.,  1667.,  1668.,  1840.,
        1911.,  1917.,  2067.,  2083.,  2200.,  2250.,  2253.,  2275.,
        2336.,  2358.,  2504.,  2840.,  2859.,  2925.,  3000.,  3369.,
        3440.,  3500.,  4196.,  5625.,  7210.,  8106., 10968.])
```
In [464]:
```
logr.predict_proba(e)[0][0]
```
Out[464]:
```
8.794009433362316e-302
```
In [465]:
```
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 [466]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
```
In [467]:
```
digits=load_digits()
digits
```
```
 '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',
```

In [468]:
```
plt.figure(figsize=(50,25))
for index,(image,label) in enumerate(zip(digits.data[0:8],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=15)
```



In [469]:
```
x_train,x_test,y_train,y_test=train_test_split(digits.data,digits.target,test_size=0
```

In [470]:
```
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(736, 64)
(1061, 64)
(736,)
(1061,)
```

In [471]:
```
logre=LogisticRegression(max_iter=10000)
logre.fit(x_train,y_train)

```

Out[471]:
```
LogisticRegression(max_iter=10000)
```

In [472]:
```
print(logre.predict(x_test))
```

```
[5 9 4 ... 4 3 4]
```

In [473]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [474]:
```
a=pd.read_csv(r"C:\USERS\user\Downloads\C8_loan-train.csv")
```

In [490]:

```
a=a.head(100)
a
```

32	LP001097	Male	No	1	Graduate	Yes	4692	0
33	LP001098	Male	Yes	0	Graduate	No	3500	1667
34	LP001100	Male	No	3+	Graduate	No	12500	3000
35	LP001106	Male	Yes	0	Graduate	No	2275	2067
36	LP001109	Male	Yes	0	Graduate	No	1828	1330
37	LP001112	Female	Yes	0	Graduate	No	3667	1459
38	LP001114	Male	No	0	Graduate	No	4166	7210
39	LP001116	Male	No	0	Not Graduate	No	3748	1668
40	LP001119	Male	No	0	Graduate	No	3600	0
41	LP001120	Male	No	0	Graduate	No	1800	1213
42	LP001123	Male	Yes	0	Graduate	No	2400	0
43	LP001131	Male	Yes	0	Graduate	No	3941	2336

In [491]:
```
b=a[[ 'ApplicantIncome', 'CoapplicantIncome','Loan_Status']]
b
```

Out[491]:

	ApplicantIncome	CoapplicantIncome	Loan_Status
0	5849	0.0	Y
1	4583	1508.0	N
2	3000	0.0	Y
3	2583	2358.0	Y
4	6000	0.0	Y
5	5417	4196.0	Y
6	2333	1516.0	Y
7	3036	2504.0	N
8	4006	1526.0	Y
9	12841	10968.0	N
10	3200	700.0	Y
11	2500	1840.0	Y
12	3073	8106.0	Y
13	1853	2840.0	N
14	1299	1086.0	Y
15	4950	0.0	Y
16	3596	0.0	Y
17	3510	0.0	N
18	4887	0.0	N
19	2600	3500.0	Y
20	7660	0.0	N
21	5955	5625.0	Y
22	2600	1911.0	N
23	3365	1917.0	N
24	3717	2925.0	N
25	9560	0.0	Y
26	2799	2253.0	Y
27	4226	1040.0	Y
28	1442	0.0	N
29	3750	2083.0	Y
30	4166	3369.0	N
31	3167	0.0	N
32	4692	0.0	N
33	3500	1667.0	Y
34	12500	3000.0	N
35	2275	2067.0	Y
36	1828	1330.0	N
37	3667	1459.0	Y
38	4166	7210.0	Y

	ApplicantIncome	CoapplicantIncome	Loan_Status
39	3748	1668.0	Y
40	3600	0.0	N
41	1800	1213.0	Y
42	2400	0.0	Y
43	3941	2336.0	Y
44	4695	0.0	Y
45	3410	0.0	Y
46	5649	0.0	Y
47	5821	0.0	Y
48	2645	3440.0	N
49	4000	2275.0	Y
50	1928	1644.0	Y
51	3086	0.0	Y
52	4230	0.0	N
53	4616	0.0	N
54	11500	0.0	N
55	2708	1167.0	Y
56	2132	1591.0	Y
57	3366	2200.0	N
58	8080	2250.0	Y
59	3357	2859.0	Y

In [492]:

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

Out[492]:

```
Y    38
N    22
Name: Loan_Status, dtype: int64
```

In [493]:

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

	ApplicantIncome	CoapplicantIncome	Loan_Status
0	5849	0.0	Y
1	4583	1508.0	N
2	3000	0.0	Y
3	2583	2358.0	Y
4	6000	0.0	Y
5	5417	4196.0	Y
6	2333	1516.0	Y
7	3036	2504.0	N
8	4006	1526.0	Y
9	12841	10968.0	N
10	3200	700.0	Y
11	2500	1840.0	Y
12	3073	8106.0	Y
13	1853	2840.0	N
14	1299	1086.0	Y
15	4950	0.0	Y
16	3596	0.0	Y
17	3510	0.0	N
18	4887	0.0	N
19	2600	3500.0	Y
20	7660	0.0	N
21	5955	5625.0	Y
22	2600	1911.0	N
23	3365	1917.0	N
24	3717	2925.0	N
25	9560	0.0	Y
26	2799	2253.0	Y
27	4226	1040.0	Y
28	1442	0.0	N
29	3750	2083.0	Y
30	4166	3369.0	N
31	3167	0.0	N
32	4692	0.0	N
33	3500	1667.0	Y
34	12500	3000.0	N
35	2275	2067.0	Y
36	1828	1330.0	N
37	3667	1459.0	Y
38	4166	7210.0	Y
39	3748	1668.0	Y
40	3600	0.0	N
41	1800	1213.0	Y
42	2400	0.0	Y
43	3941	2336.0	Y
44	4695	0.0	Y
45	3410	0.0	Y
46	5649	0.0	Y
47	5821	0.0	Y
48	2645	3440.0	N
49	4000	2275.0	Y
50	1928	1644.0	Y
51	3086	0.0	Y
52	4230	0.0	N
53	4616	0.0	N
54	11500	0.0	N
55	2708	1167.0	Y
56	2132	1591.0	Y
57	3366	2200.0	N
58	8080	2250.0	Y
59	3357	2859.0	Y

In [494]:
```
g1={"Loan_Status":{'g1':1}}
a=a.replace(g1)
print(a)
```

```
      Loan_ID  Gender Married Dependents     Education Self_Employed  \
0    LP001002    Male      No          0      Graduate            No
1    LP001003    Male     Yes          1      Graduate            No
2    LP001005    Male     Yes          0      Graduate           Yes
3    LP001006    Male     Yes          0  Not Graduate            No
4    LP001008    Male      No          0      Graduate            No
5    LP001011    Male     Yes          2      Graduate           Yes
6    LP001013    Male     Yes          0  Not Graduate            No
7    LP001014    Male     Yes         3+      Graduate            No
8    LP001018    Male     Yes          2      Graduate            No
9    LP001020    Male     Yes          1      Graduate            No
10   LP001024    Male     Yes          2      Graduate            No
11   LP001027    Male     Yes          2      Graduate           NaN
12   LP001028    Male     Yes          2      Graduate            No
13   LP001029    Male      No          0      Graduate            No
14   LP001030    Male     Yes          2      Graduate            No
15   LP001032    Male      No          0      Graduate            No
16   LP001034    Male      No          1  Not Graduate            No
17   LP001036  Female      No          0      Graduate            No
```

In [495]:
```
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 [496]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [497]:
```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[497]:
```
RandomForestClassifier()
```

In [498]:
```
parameters={'max_depth':[1,2,3,4,5],
            'min_samples_leaf':[5,10,15,20,25],
            'n_estimators':[10,20,30,40,50]}
```

In [499]:
```
from sklearn.model_selection import GridSearchCV
```

In [500]:
```
grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy"
grid_search.fit(x_train,y_train)
```

Out[500]:
```
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 [501]:
```
grid_search.best_score_
```

Out[501]:
```
0.7142857142857143
```

In [502]:
```
rfc_best=grid_search.best_estimator_
```

In [503]:

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

In [504]:

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

Out[504]:

```
[Text(558.0, 407.70000000000005, 'CoapplicantIncome <= 543.0\ngini = 0.444\nsamples = 2
9\nvalue = [14, 28]\nclass = No'),
 Text(279.0, 135.89999999999998, 'gini = 0.498\nsamples = 11\nvalue = [9, 8]\nclass = Y
es'),
 Text(837.0, 135.89999999999998, 'gini = 0.32\nsamples = 18\nvalue = [5, 20]\nclass = N
o')]
```