20115015

C1

```
import pandas as pd
from sklearn.linear_model import LogisticRegression
import matplotlib.pyplot as plt


from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split


df=pd.read_csv("1_ionosphere.csv")
df
```

	1	0	0.99539	-0.05889	0.85243	0.02306	0.83398	-0.37708	1.1	0.03760
0	1	0	1.00000	-0.18829	0.93035	-0.36156	-0.10868	-0.93597	1.00000	-0.04549
1	1	0	1.00000	-0.03365	1.00000	0.00485	1.00000	-0.12062	0.88965	0.01198
2	1	0	1.00000	-0.45161	1.00000	1.00000	0.71216	-1.00000	0.00000	0.00000
3	1	0	1.00000	-0.02401	0.94140	0.06531	0.92106	-0.23255	0.77152	-0.16399
4	1	0	0.02337	-0.00592	-0.09924	-0.11949	-0.00763	-0.11824	0.14706	0.06637
...	...	...	...	...	...	...	...	...	...	...
345	1	0	0.83508	0.08298	0.73739	-0.14706	0.84349	-0.05567	0.90441	-0.04622
346	1	0	0.95113	0.00419	0.95183	-0.02723	0.93438	-0.01920	0.94590	0.01606
347	1	0	0.94701	-0.00034	0.93207	-0.03227	0.95177	-0.03431	0.95584	0.02446
348	1	0	0.90608	-0.01657	0.98122	-0.01989	0.95691	-0.03646	0.85746	0.00110
349	1	0	0.84710	0.13533	0.73638	-0.06151	0.87873	0.08260	0.88928	-0.09139

350 rows × 35 columns

```
x=df.iloc[:,:10]
y=df.iloc[:,-1]
f=StandardScaler().fit_transform(x)
lo=LogisticRegression()
lo.fit(f,y)
```

```
LogisticRegression()
```

```
val=[[6,45,234,534,6,3456,345,4,53,45]]
lo.predict(val)
```

```
array(['g'], dtype=object)
```

```
lo.score(f,y)
```

```
0.8885714285714286
```

```
df["g"].value_counts()
```

```
g    224
b    126
Name: g, dtype: int64
```

```
g1={"g":{'g':1,'b':2}}
df=df.replace(g1)
df
```

	1	0	0.99539	-0.05889	0.85243	0.02306	0.83398	-0.37708	1.1	0.037
0	1	0	1.00000	-0.18829	0.93035	-0.36156	-0.10868	-0.93597	1.00000	-0.045
1	1	0	1.00000	-0.03365	1.00000	0.00485	1.00000	-0.12062	0.88965	0.011
2	1	0	1.00000	-0.45161	1.00000	1.00000	0.71216	-1.00000	0.00000	0.000
3	1	0	1.00000	-0.02401	0.94140	0.06531	0.92106	-0.23255	0.77152	-0.163
4	1	0	0.02337	-0.00592	-0.09924	-0.11949	-0.00763	-0.11824	0.14706	0.066
...	...	...	...	...	...	...	...	...	...	
345	1	0	0.83508	0.08298	0.73739	-0.14706	0.84349	-0.05567	0.90441	-0.046

```
x1=df.drop(["g"],axis=1)
y1=df["g"]
x_train,x_test,y_train,y_test=train_test_split(x1,y1,test_size=0.3)
```

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

```
RandomForestClassifier()
```

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

```
from sklearn.model_selection import GridSearchCV

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

```
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')
```

```
grid_search.best_score_
```

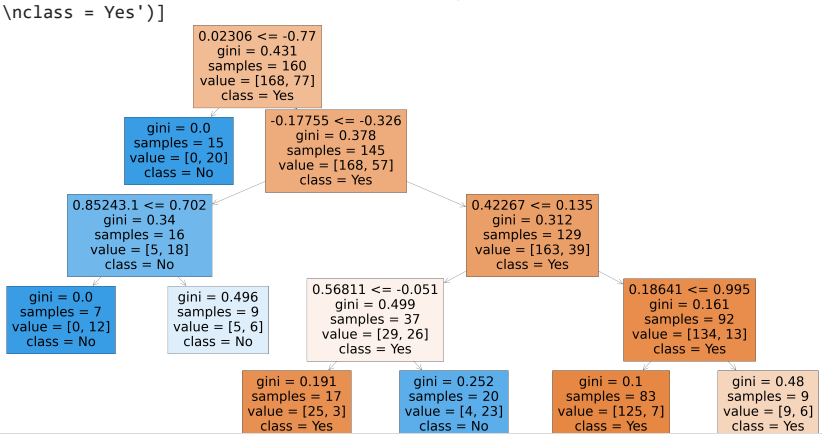
```
0.9265293882447021
```

```
rfc_best=grid_search.best_estimator_
```

```
from sklearn.tree import plot_tree

plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x1.columns,class_names=['Yes','No'],filled=True)
```

```
[Text(1420.3636363636363, 1956.96, '0.02306 <= -0.77\ngini = 0.431\nsamples =
160\nvalue = [168, 77]\nclass = Yes'),
 Text(1014.5454545454545, 1522.0800000000002, 'gini = 0.0\nsamples = 15\nvalue = [0,
20]\nclass = No'),
 Text(1826.1818181818182, 1522.0800000000002, '-0.17755 <= -0.326\ngini =
0.378\nsamples = 145\nvalue = [168, 57]\nclass = Yes'),
 Text(811.6363636363636, 1087.2, '0.85243.1 <= 0.702\ngini = 0.34\nsamples =
16\nvalue = [5, 18]\nclass = No'),
 Text(405.8181818181818, 652.3200000000002, 'gini = 0.0\nsamples = 7\nvalue = [0,
12]\nclass = No'),
 Text(1217.4545454545455, 652.3200000000002, 'gini = 0.496\nsamples = 9\nvalue = [5,
6]\nclass = No'),
 Text(2840.7272727272725, 1087.2, '0.42267 <= 0.135\ngini = 0.312\nsamples =
129\nvalue = [163, 39]\nclass = Yes'),
 Text(2029.090909090909, 652.3200000000002, '0.56811 <= -0.051\ngini =
[23, 3]\nclass = Yes'),
 Text(2434.909090909091, 217.44000000000005, 'gini = 0.252\nsamples = 20\nvalue =
[4, 23]\nclass = No'),
 Text(3652.3636363636365, 652.3200000000002, '0.18641 <= 0.995\ngini =
0.161\nsamples = 92\nvalue = [134, 13]\nclass = Yes'),
 Text(3246.5454545454545, 217.44000000000005, 'gini = 0.1\nsamples = 83\nvalue =
[125, 7]\nclass = Yes'),
 Text(4058.181818181818, 217.44000000000005, 'gini = 0.48\nsamples = 9\nvalue = [9,
6]\nclass = Yes')]
```



C2

```
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

+ Code + Text

```
from sklearn.ensemble import RandomForestClassifier
import matplotlib.pyplot as plt
from sklearn.model_selection import GridSearchCV
from sklearn.tree import plot_tree


df=pd.read_csv("C2_test.gender_submission.csv")
df1=pd.read_csv("C2_train.gender_submission.csv")


df_=df.drop(["Cabin","Name","Embarked","Ticket","PassengerId","Sex"],axis=1)
df1_=df1.drop(["Survived","Cabin","Name","Embarked","Ticket","PassengerId"],axis=1)
print(df_)
print(df1_)
```

```
     Pclass   Age  SibSp  Parch      Fare
0         3  34.5      0      0    7.8292
1         3  47.0      1      0    7.0000
2         2  62.0      0      0    9.6875
3         3  27.0      0      0    8.6625
4         3  22.0      1      1   12.2875
..      ...   ...    ...    ...       ...
413       3   NaN      0      0    8.0500
414       1  39.0      0      0  108.9000
415       3  38.5      0      0    7.2500
416       3   NaN      0      0    8.0500
417       3   NaN      1      1   22.3583

[418 rows x 5 columns]
     Pclass     Sex   Age  SibSp  Parch     Fare
0         3    male  22.0      1      0   7.2500
1         1  female  38.0      1      0  71.2833
2         3  female  26.0      0      0   7.9250
3         1  female  35.0      1      0  53.1000
4         3    male  35.0      0      0   8.0500
..      ...     ...   ...    ...    ...      ...
886       2    male  27.0      0      0  13.0000
887       1  female  19.0      0      0  30.0000
888       3  female   NaN      1      2  23.4500
889       1    male  26.0      0      0  30.0000
890       3    male  32.0      0      0   7.7500

[891 rows x 6 columns]
```

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

```
df_=df_.dropna()
df1_=df1_.dropna()
df1_.info()
df_.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 714 entries, 0 to 890
Data columns (total 6 columns):
```

```
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Pclass  714 non-null    int64
 1   Sex     714 non-null    object
 2   Age     714 non-null    float64
 3   SibSp   714 non-null    int64
 4   Parch   714 non-null    int64
 5   Fare    714 non-null    float64
dtypes: float64(2), int64(3), object(1)
memory usage: 39.0+ KB
<class 'pandas.core.frame.DataFrame'>
Int64Index: 331 entries, 0 to 415
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Pclass  331 non-null    int64
 1   Age     331 non-null    float64
 2   SibSp   331 non-null    int64
 3   Parch   331 non-null    int64
 4   Fare    331 non-null    float64
dtypes: float64(2), int64(3)
memory usage: 15.5 KB
```

```
y=df1_["Sex"]
x=df1_.drop(["Sex"],axis=1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

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

```
RandomForestClassifier()
```

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

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

```
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')
```

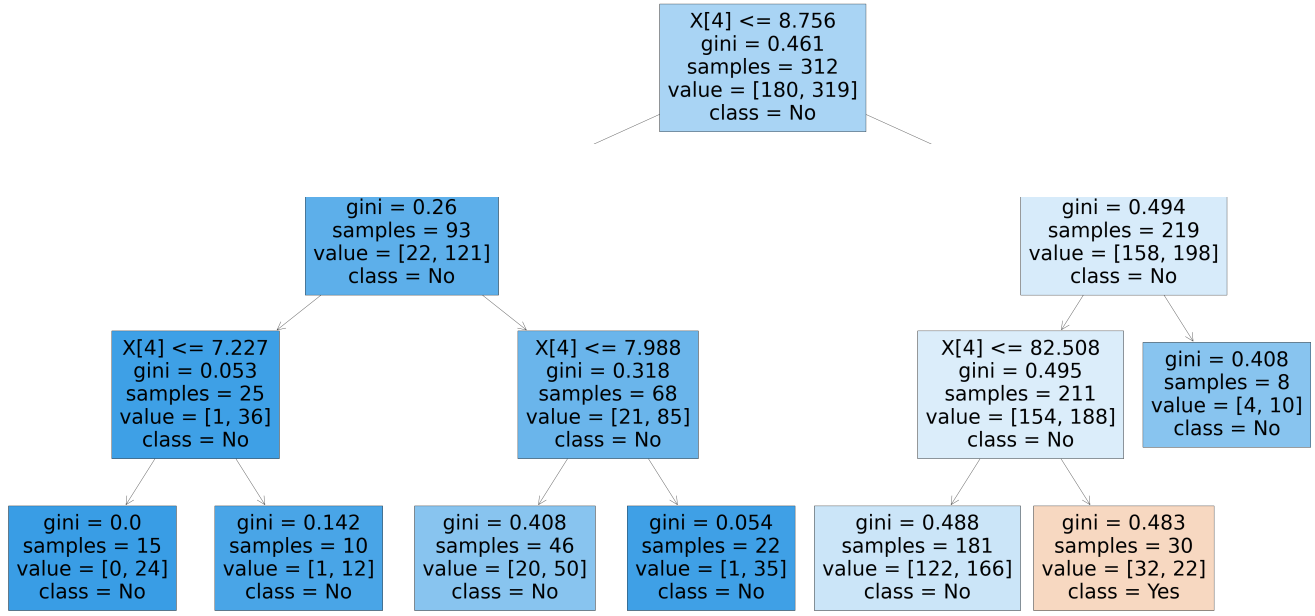
```
grid_search.best_score_
```

```
0.6653654618473896
```

```
rfc_best=grid_search.best_estimator_
```

```
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],class_names=['Yes','No','Yes'],filled=True)
```

```
[Text(2575.3846153846152, 1902.6000000000001, 'X[4] <= 8.756\ngini = 0.461\nsamples = 312\nvalue = [180, 319]\nclass = No'),
 Text(1373.5384615384614, 1359.0, 'X[4] <= 7.746\ngini = 0.26\nsamples = 93\nvalue = [22, 121]\nclass = No'),
 Text(686.7692307692307, 815.4000000000001, 'X[4] <= 7.227\ngini = 0.053\nsamples = 25\nvalue = [1, 36]\nclass = No'),
 Text(343.38461538461536, 271.79999999999995, 'gini = 0.0\nsamples = 15\nvalue = [0, 24]\nclass = No'),
 Text(1030.1538461538462, 271.79999999999995, 'gini = 0.142\nsamples = 10\nvalue = [1, 12]\nclass = No'),
 Text(2060.3076923076924, 815.4000000000001, 'X[4] <= 7.988\ngini = 0.318\nsamples = 68\nvalue = [21, 85]\nclass = No'),
 Text(1716.9230769230767, 271.79999999999995, 'gini = 0.408\nsamples = 46\nvalue = [20, 50]\nclass = No'),
 Text(2403.6923076923076, 271.79999999999995, 'gini = 0.054\nsamples = 22\nvalue = [1, 35]\nclass = No'),
 Text(3777.230769230769, 1359.0, 'X[2] <= 3.5\ngini = 0.494\nsamples = 219\nvalue = [158, 198]\nclass = No'),
 Text(3433.8461538461534, 815.4000000000001, 'X[4] <= 82.508\ngini = 0.495\nsamples = 211\nvalue = [154, 188]\nclass = No'),
 Text(3090.461538461538, 271.79999999999995, 'gini = 0.488\nsamples = 181\nvalue = [122, 166]\nclass = No'),
 Text(3777.230769230769, 271.79999999999995, 'gini = 0.483\nsamples = 30\nvalue = [32, 22]\nclass = Yes'),
 Text(4120.615384615385, 815.4000000000001, 'gini = 0.408\nsamples = 8\nvalue = [4, 10]\nclass = No')]
```



C3

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split


from sklearn.ensemble import RandomForestClassifier
import matplotlib.pyplot as plt
from sklearn.model_selection import GridSearchCV


df=pd.read_csv("C3_bot_detection_data.csv")
df
```

	User ID	Username	Tweet	Retweet Count	Mention Count	Follower Count	Verified	La
0	132131	flong	Station activity person against natural majori...	85	1	2353	False	
1	289683	hinesstephanie	Authority research natural life material staff...	55	5	9617	True	
2	779715	roberttran	Manage whose quickly especially foot none to g...	6	2	4363	True	
3	696168	pmason	Just cover eight opportunity strong policy which.	54	5	2242	True	
4	704441	noah87	Animal sign six data good or.	26	3	8438	False	
...	...	...	...	...	...	...	...	

```
df1=df.iloc[:,3:8]
df1
```

	Retweet Count	Mention Count	Follower Count	Verified	Bot Label
0	85	1	2353	False	1
1	55	5	9617	True	0
2	6	2	4363	True	0
3	54	5	2242	True	1
4	26	3	8438	False	1
...	...	...	...	...	...
49995	64	0	9911	True	1
49996	18	5	9900	False	1
49997	43	3	6313	True	1
49998	45	1	6343	False	0
49999	91	4	4006	False	0

50000 rows × 5 columns

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 5 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Retweet Count   50000 non-null  int64
 1   Mention Count   50000 non-null  int64
 2   Follower Count  50000 non-null  int64
 3   Verified        50000 non-null  bool
 4   Bot Label       50000 non-null  int64
dtypes: bool(1), int64(4)
memory usage: 1.6 MB
```

```
y=df1["Verified"]
x=df1.drop(["Verified"],axis=1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

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

```
RandomForestClassifier()
```

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

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

```
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')
```

```
grid_search.best_score_
```

```
0.5039714285714285
```

```
rfc_best=grid_search.best_estimator_
```

```
from sklearn.tree import plot_tree

plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','No'],filled=True)
```

```
[Text(2311.714285714286, 1993.2, 'Follower Count <= 267.5\ngini = 0.5\nsamples = 22141\nvalue = [17342, 17658]\nclass = No'),
 Text(1307.3142857142857, 1630.8000000000002, 'Retweet Count <= 83.5\ngini = 0.497\nsamples = 602\nvalue = [512, 437]\nclass = Yes'),
 Text(573.9428571428572, 1268.4, 'Retweet Count <= 5.5\ngini = 0.492\nsamples = 501\nvalue = [445, 343]\nclass = Yes'),
 Text(255.0857142857143, 906.0, 'Follower Count <= 126.0\ngini = 0.41\nsamples = 36\nvalue = [42, 17]\nclass = Yes'),
 Text(127.54285714285714, 543.5999999999999, 'gini = 0.236\nsamples = 15\nvalue = [19, 3]\nclass = Yes'),
 Text(382.62857142857143, 543.5999999999999, 'gini = 0.47\nsamples = 21\nvalue = [23, 14]\nclass = Yes'),
 Text(892.8, 906.0, 'Follower Count <= 98.5\ngini = 0.494\nsamples = 465\nvalue = [403, 326]\nclass = Yes'),
 Text(637.7142857142858, 543.5999999999999, 'Follower Count <= 62.5\ngini = 0.496\nsamples = 174\nvalue = [123, 148]\nclass = No'),
 Text(510.1714285714286, 181.19999999999982, 'gini = 0.499\nsamples = 110\nvalue = [93, 84]\nclass = Yes'),
 Text(765.2571428571429, 181.19999999999982, 'gini = 0.435\nsamples = 64\nvalue = [30, 64]\nclass = No'),
 Text(1147.8857142857144, 543.5999999999999, 'Follower Count <= 107.5\ngini = 0.475\nsamples = 291\nvalue = [280, 178]\nclass = Yes'),
 Text(1020.3428571428572, 181.19999999999982, 'gini = 0.251\nsamples = 17\nvalue = [29, 5]\nclass = Yes'),
 Text(1275.4285714285716, 181.19999999999982, 'gini = 0.483\nsamples = 274\nvalue = [251, 173]\nclass = Yes'),
 Text(2040.6857142857143, 1268.4, 'Retweet Count <= 97.5\ngini = 0.486\nsamples = 101\nvalue = [67, 94]\nclass = No'),
 Text(1913.142857142857, 906.0, 'Follower Count <= 120.5\ngini = 0.471\nsamples = 83\nvalue = [50, 82]\nclass = No'),
 Text(1658.057142857143, 543.5999999999999, 'Mention Count <= 2.5\ngini = 0.393\nsamples = 33\nvalue = [14, 38]\nclass = No'),
 Text(1530.5142857142857, 181.19999999999982, 'gini = 0.452\nsamples = 17\nvalue = [10, 19]\nclass = No'),
 Text(1785.6, 181.19999999999982, 'gini = 0.287\nsamples = 16\nvalue = [4, 19]\nclass = No'),
 Text(2168.2285714285713, 543.5999999999999, 'Mention Count <= 2.5\ngini = 0.495\nsamples = 50\nvalue = [36, 44]\nclass = No'),
 Text(2040.6857142857143, 181.19999999999982, 'gini = 0.5\nsamples = 28\nvalue = [24, 23]\nclass = Yes'),
 Text(2295.7714285714287, 181.19999999999982, 'gini = 0.463\nsamples = 22\nvalue = [12, 21]\nclass = No'),
 Text(2168.2285714285713, 906.0, 'gini = 0.485\nsamples = 18\nvalue = [17, 12]\nclass = Yes'),
 Text(3316.114285714286, 1630.8000000000002, 'Mention Count <= 1.5\ngini = 0.5\nsamples = 21539\nvalue = [16830, 17221]\nclass = No'),
 Text(2805.942857142857, 1268.4, 'Follower Count <= 312.0\ngini = 0.5\nsamples = 7171\nvalue = [5707, 5571]\nclass = Yes'),
 Text(2678.4, 906.0, 'gini = 0.454\nsamples = 28\nvalue = [15, 28]\nclass = No'),
 Text(2933.4857142857145, 906.0, 'Follower Count <= 360.5\ngini = 0.5\nsamples = 7143\nvalue = [5692, 5543]\nclass = Yes'),
 Text(2678.4, 543.5999999999999, 'Retweet Count <= 62.5\ngini = 0.465\nsamples = 43\nvalue = [43, 25]\nclass = Yes'),
 Text(2550.857142857143, 181.19999999999982, 'gini = 0.5\nsamples = 25\nvalue = [18, 18]\nclass = Yes'),
 Text(2805.942857142857, 181.19999999999982, 'gini = 0.342\nsamples = 18\nvalue = [25, 7]\nclass = Yes'),
 Text(3188.5714285714284, 543.5999999999999, 'Retweet Count <= 84.5\ngini = 0.5\nsamples = 7100\nvalue = [5649, 5518]\nclass = Yes'),
 Text(3061.0285714285715, 181.19999999999982, 'gini = 0.5\nsamples = 5954\nvalue = [4807, 4576]\nclass = Yes'),
 Text(3316.114285714286, 181.19999999999982, 'gini = 0.498\nsamples = 1146\nvalue = [842, 942]\nclass = No'),
 Text(3826.285714285714, 1268.4, 'Follower Count <= 279.5\ngini = 0.5\nsamples = 14368\nvalue = [11123, 11650]\nclass = No'),
 Text(3698.7428571428572, 906.0, 'gini = 0.35\nsamples = 20\nvalue = [7, 24]\nclass = No'),
 Text(3953.8285714285716, 906.0, 'Mention Count <= 2.5\ngini = 0.5\nsamples = 14348\nvalue = [11116, 11626]\nclass = No'),
 Text(3698.7428571428572, 543.5999999999999, 'Follower Count <= 6726.0\ngini = 0.5\nsamples = 3512\nvalue = [2733, 2788]\nclass
```

C4

```
df2=pd.read_csv("C4_framingham.csv")
df2
```

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI
0	1	39	4.0	0	0.0	0.0	0	0	0	195.0	106.0	70.0	26.97
1	0	46	2.0	0	0.0	0.0	0	0	0	250.0	121.0	81.0	28.73
2	1	48	1.0	1	20.0	0.0	0	0	0	245.0	127.5	80.0	25.34
3	0	61	3.0	1	30.0	0.0	0	1	0	225.0	150.0	95.0	28.58
4	0	46	3.0	1	23.0	0.0	0	0	0	285.0	130.0	84.0	23.10
...	...	...	...	...	...	...	...	...	...	...	...	...	...
4233	1	50	1.0	1	1.0	0.0	0	1	0	313.0	179.0	92.0	25.97
4234	1	51	3.0	1	43.0	0.0	0	0	0	207.0	126.5	80.0	19.71
4235	0	48	2.0	1	20.0	NaN	0	0	0	248.0	131.0	72.0	22.00
4236	0	44	1.0	1	15.0	0.0	0	0	0	210.0	126.5	87.0	19.16
4237	0	52	2.0	0	0.0	0.0	0	0	0	269.0	133.5	83.0	21.47

4238 rows × 16 columns

```
df2=df2.dropna()
```

```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3656 entries, 0 to 4237
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   male             3656 non-null   int64  
 1   age              3656 non-null   int64  
 2   education        3656 non-null   float64
 3   currentSmoker    3656 non-null   int64
```

```
 4   cigsPerDay       3656 non-null   float64
 5   BPMeds           3656 non-null   float64
 6   prevalentStroke  3656 non-null   int64
 7   prevalentHyp     3656 non-null   int64
 8   diabetes         3656 non-null   int64
 9   totChol          3656 non-null   float64
 10  sysBP            3656 non-null   float64
 11  diaBP            3656 non-null   float64
 12  BMI              3656 non-null   float64
 13  heartRate        3656 non-null   float64
 14  glucose          3656 non-null   float64
 15  TenYearCHD       3656 non-null   int64
dtypes: float64(9), int64(7)
memory usage: 485.6 KB
```

```
y=df2["diabetes"]
x=df2.drop(["diabetes"],axis=1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

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

```
RandomForestClassifier()
```

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

```
from sklearn.model_selection import GridSearchCV

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

```
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')
```

```
grid_search.best_score_
```

```
0.980851739640344
```

```
rfc_best=grid_search.best_estimator_
```

```
from sklearn.tree import plot_tree

plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','No'],filled=True)
```

```
[Text(2996.666666666667, 1993.2, 'BPMeds <= 0.5\ngini = 0.055\nsamples =
1616\nvalue = [2487, 72]\nclass = Yes'),
 Text(1860.0, 1630.8000000000002, 'age <= 61.5\ngini = 0.048\nsamples =
1563\nvalue = [2412, 61]\nclass = Yes'),
 Text(909.3333333333334, 1268.4, 'male <= 0.5\ngini = 0.034\nsamples = 1395\nvalue
= [2178, 38]\nclass = Yes'),
 Text(496.0, 906.0, 'glucose <= 133.5\ngini = 0.013\nsamples = 768\nvalue = [1202,
8]\nclass = Yes'),
 Text(330.6666666666667, 543.5999999999999, 'heartRate <= 74.5\ngini =
0.005\nsamples = 762\nvalue = [1201, 3]\nclass = Yes'),
 Text(165.33333333333334, 181.19999999999982, 'gini = 0.0\nsamples = 311\nvalue =
[461, 0]\nclass = Yes'),
 Text(496.0, 181.19999999999982, 'gini = 0.008\nsamples = 451\nvalue = [740,
3]\nclass = Yes'),
 Text(661.3333333333334, 543.5999999999999, 'gini = 0.278\nsamples = 6\nvalue =
[1, 5]\nclass = No'),
 Text(1322.6666666666667, 906.0, 'sysBP <= 156.75\ngini = 0.058\nsamples =
627\nvalue = [976, 30]\nclass = Yes'),
 Text(992.0, 543.5999999999999, 'glucose <= 143.0\ngini = 0.045\nsamples =
566\nvalue = [894, 21]\nclass = Yes'),
 Text(826.6666666666667, 181.19999999999982, 'gini = 0.018\nsamples = 561\nvalue =
[894, 8]\nclass = Yes'),
 Text(1157.3333333333335, 181.19999999999982, 'gini = 0.0\nsamples = 5\nvalue =
[0, 13]\nclass = No'),
 Text(1653.3333333333335, 543.5999999999999, 'currentSmoker <= 0.5\ngini =
0.178\nsamples = 61\nvalue = [82, 9]\nclass = Yes'),
 Text(1488.0, 181.19999999999982, 'gini = 0.384\nsamples = 19\nvalue = [20,
7]\nclass = Yes'),
 Text(1818.6666666666667, 181.19999999999982, 'gini = 0.061\nsamples = 42\nvalue =
[62, 2]\nclass = Yes'),
 Text(2810.666666666667, 1268.4, 'glucose <= 108.5\ngini = 0.163\nsamples =
168\nvalue = [234, 23]\nclass = Yes'),
 Text(2480.0, 906.0, 'glucose <= 81.5\ngini = 0.017\nsamples = 153\nvalue = [229,
2]\nclass = Yes'),
 Text(2314.666666666667, 543.5999999999999, 'prevalentHyp <= 0.5\ngini =
0.032\nsamples = 81\nvalue = [120, 2]\nclass = Yes'),
```

C5

```
 Text(2845.333333333333, 543.5999999999999, 'gini = 0.0\nsamples = 72\nvalue =
```

```
df3=pd.read_csv("C5_health care diabetes.csv")
df3
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

768 rows × 9 columns

```
df3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

```
y1=df3["Outcome"]
x1=df3.drop(["Outcome"],axis=1)
x1_train,x1_test,y1_train,y1_test=train_test_split(x1,y1,test_size=0.3)
```

```
rfc=RandomForestClassifier()
rfc.fit(x1_train,y1_train)
```

```
RandomForestClassifier()
```

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

```
grid_search = GridSearchCV(estimator=rfc,param_grid=parameter,cv=2,scoring="accuracy")
grid_search.fit(x1_train,y1_train)
```

```
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')
```

```
grid_search.best_score_
```

```
0.7635243855074072
```

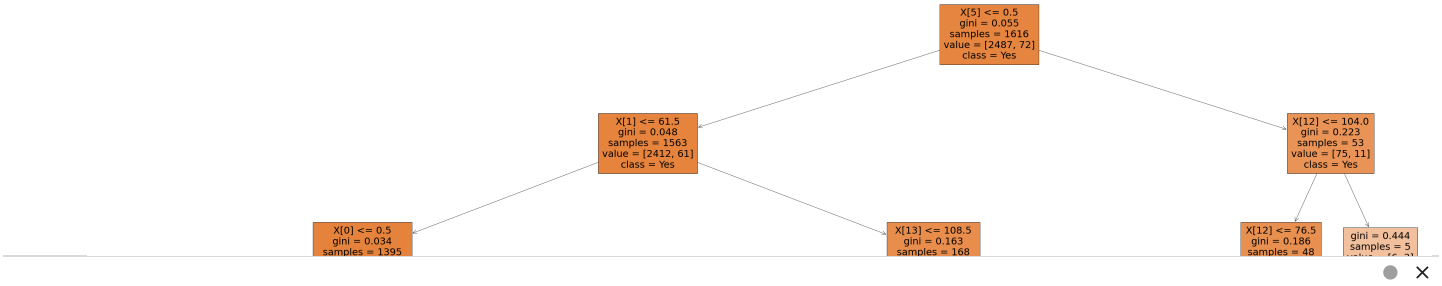
```
len(x1.columns)
```

```
8
```

```
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],class_names=['Yes','No'],filled=True)
```

```
[Text(2996.666666666667, 1993.2, 'X[5] <= 0.5\ngini = 0.055\nsamples = 1616\nvalue = [2487, 72]\nclass = Yes'),
 Text(1860.0, 1630.8000000000002, 'X[1] <= 61.5\ngini = 0.048\nsamples = 1563\nvalue = [2412, 61]\nclass = Yes'),
 Text(909.3333333333334, 1268.4, 'X[0] <= 0.5\ngini = 0.034\nsamples = 1395\nvalue = [2178, 38]\nclass = Yes'),
 Text(496.0, 906.0, 'X[13] <= 133.5\ngini = 0.013\nsamples = 768\nvalue = [1202, 8]\nclass = Yes'),
 Text(330.6666666666667, 543.5999999999999, 'X[12] <= 74.5\ngini = 0.005\nsamples = 762\nvalue = [1201, 3]\nclass = Yes'),
 Text(165.33333333333334, 181.19999999999982, 'gini = 0.0\nsamples = 311\nvalue = [461, 0]\nclass = Yes'),
 Text(496.0, 181.19999999999982, 'gini = 0.008\nsamples = 451\nvalue = [740, 3]\nclass = Yes'),
 Text(661.3333333333334, 543.5999999999999, 'gini = 0.278\nsamples = 6\nvalue = [1, 5]\nclass = No'),
 Text(1322.6666666666667, 906.0, 'X[9] <= 156.75\ngini = 0.058\nsamples = 627\nvalue = [976, 30]\nclass = Yes'),
 Text(992.0, 543.5999999999999, 'X[13] <= 143.0\ngini = 0.045\nsamples = 566\nvalue = [894, 21]\nclass = Yes'),
 Text(826.6666666666667, 181.19999999999982, 'gini = 0.018\nsamples = 561\nvalue = [894, 8]\nclass = Yes'),
 Text(1157.3333333333335, 181.19999999999982, 'gini = 0.0\nsamples = 5\nvalue = [0, 13]\nclass = No'),
 Text(1653.3333333333335, 543.5999999999999, 'X[3] <= 0.5\ngini = 0.178\nsamples = 61\nvalue = [82, 9]\nclass = Yes'),
 Text(1488.0, 181.19999999999982, 'gini = 0.384\nsamples = 19\nvalue = [20, 7]\nclass = Yes'),
 Text(1818.6666666666667, 181.19999999999982, 'gini = 0.061\nsamples = 42\nvalue = [62, 2]\nclass = Yes'),
 Text(2810.666666666667, 1268.4, 'X[13] <= 108.5\ngini = 0.163\nsamples = 168\nvalue = [234, 23]\nclass = Yes'),
 Text(2480.0, 906.0, 'X[13] <= 81.5\ngini = 0.017\nsamples = 153\nvalue = [229, 2]\nclass = Yes'),
 Text(2314.666666666667, 543.5999999999999, 'X[7] <= 0.5\ngini = 0.032\nsamples = 81\nvalue = [120, 2]\nclass = Yes'),
 Text(2149.3333333333335, 181.19999999999982, 'gini = 0.0\nsamples = 42\nvalue = [67, 0]\nclass = Yes'),


 Text(3141.3333333333335, 906.0, 'X[9] <= 137.0\ngini = 0.311\nsamples = 15\nvalue = [5, 21]\nclass = No'),
 Text(2976.0, 543.5999999999999, 'gini = 0.42\nsamples = 7\nvalue = [3, 7]\nclass = No'),
 Text(3306.666666666667, 543.5999999999999, 'gini = 0.219\nsamples = 8\nvalue = [2, 14]\nclass = No'),
 Text(4133.333333333334, 1630.8000000000002, 'X[12] <= 104.0\ngini = 0.223\nsamples = 53\nvalue = [75, 11]\nclass = Yes'),
 Text(3968.0, 1268.4, 'X[12] <= 76.5\ngini = 0.186\nsamples = 48\nvalue = [69, 8]\nclass = Yes'),
 Text(3802.666666666667, 906.0, 'X[11] <= 27.725\ngini = 0.256\nsamples = 30\nvalue = [45, 8]\nclass = Yes'),
 Text(3637.3333333333335, 543.5999999999999, 'gini = 0.0\nsamples = 18\nvalue = [33, 0]\nclass = Yes'),
 Text(3968.0, 543.5999999999999, 'X[8] <= 276.5\ngini = 0.48\nsamples = 12\nvalue = [12, 8]\nclass = Yes'),
 Text(3802.666666666667, 181.19999999999982, 'gini = 0.5\nsamples = 5\nvalue = [5, 5]\nclass = Yes'),
 Text(4133.333333333334, 181.19999999999982, 'gini = 0.42\nsamples = 7\nvalue = [7, 3]\nclass = Yes'),
 Text(4133.333333333334, 906.0, 'gini = 0.0\nsamples = 18\nvalue = [24, 0]\nclass = Yes'),
 Text(4298.666666666667, 1268.4, 'gini = 0.444\nsamples = 5\nvalue = [6, 3]\nclass = Yes')]
```



C6

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression,LinearRegression
from sklearn.model_selection import train_test_split
from numpy import cov


from sklearn.ensemble import RandomForestClassifier
import matplotlib.pyplot as plt
from sklearn.model_selection import GridSearchCV
from sklearn.tree import plot_tree


df=pd.read_csv("C6_bmi.csv")
df
```

	Gender	Height	Weight	Index
0	Male	174	96	4
1	Male	189	87	2
2	Female	185	110	4
3	Female	195	104	3

495	Female	150	153	5
496	Female	184	121	4
497	Female	141	136	5
498	Male	150	95	5
499	Male	173	131	5

500 rows × 4 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Gender  500 non-null    object
 1   Height  500 non-null    int64
 2   Weight  500 non-null    int64
 3   Index   500 non-null    int64
dtypes: int64(3), object(1)
memory usage: 15.8+ KB
```

```
y=df["Gender"]
x=df.drop(["Gender"],axis=1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)


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

```
RandomForestClassifier()
```

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


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

```
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')
```

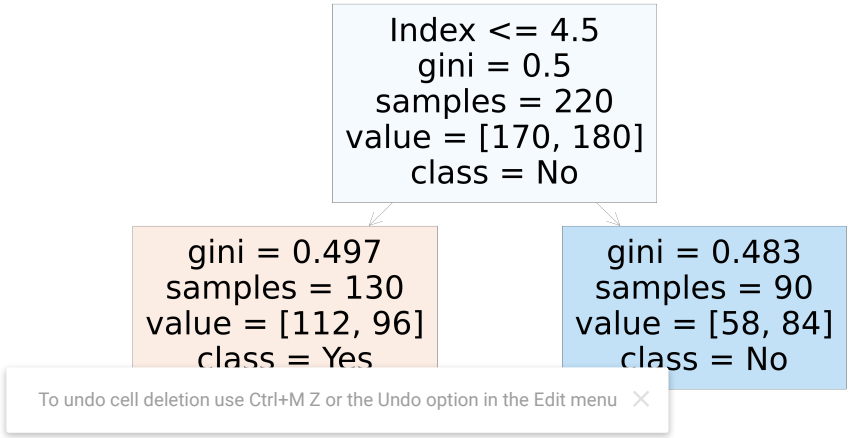
```
grid_search.best_score_
```

```
0.5685714285714285
```

```
rfc_best=grid_search.best_estimator_
```

```
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','No'],filled=True)
```

```
[Text(2232.0, 1630.8000000000002, 'Index <= 4.5\ngini = 0.5\nsamples = 220\nvalue
= [170, 180]\nclass = No'),
 Text(1116.0, 543.5999999999999, 'gini = 0.497\nsamples = 130\nvalue = [112,
96]\nclass = Yes'),
 Text(3348.0, 543.5999999999999, 'gini = 0.483\nsamples = 90\nvalue = [58,
84]\nclass = No')]
```



To undo cell deletion use Ctrl+M Z or the Undo option in the Edit menu

C7

```
df1=pd.read_csv("c7_used_cars.csv")
df1
```

	Unnamed: 0	model	year	price	transmission	mileage	fuelType	tax	mpg	en
0	0	T-Roc	2019	25000	Automatic	13904	Diesel	145	49.6	
1	1	T-Roc	2019	26883	Automatic	4562	Diesel	145	49.6	
2	2	T-Roc	2019	20000	Manual	7414	Diesel	145	50.4	
3	3	T-Roc	2019	33492	Automatic	4825	Petrol	145	32.5	
4	4	T-Roc	2019	22900	Semi-Auto	6500	Petrol	150	39.8	
...	...	...	...	...	...	...	...	...	...	...
99182	10663	A3	2020	16999	Manual	4018	Petrol	145	49.6	
99183	10664	A3	2020	16999	Manual	1978	Petrol	150	49.6	
99184	10665	A3	2020	17199	Manual	609	Petrol	150	49.6	
99185	10666	Q3	2017	19499	Automatic	8646	Petrol	150	47.9	
99186	10667	Q3	2016	15999	Manual	11855	Petrol	150	47.9	

```
df2=df1.drop(["transmission","Make","model","Unnamed: 0"],axis=1)
df2
```

	year	price	mileage	fuelType	tax	mpg	engineSize
0	2019	25000	13904	Diesel	145	49.6	2.0
1	2019	26883	4562	Diesel	145	49.6	2.0
2	2019	20000	7414	Diesel	145	50.4	2.0
3	2019	33492	4825	Petrol	145	32.5	2.0
4	2019	22900	6500	Petrol	150	39.8	1.5
...	...	...	...	...	...	...	...
99182	2020	16999	4018	Petrol	145	49.6	1.0
99183	2020	16999	1978	Petrol	150	49.6	1.0
99184	2020	17199	609	Petrol	150	49.6	1.0
99185	2017	19499	8646	Petrol	150	47.9	1.4
99186	2016	15999	11855	Petrol	150	47.9	1.4

```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99187 entries, 0 to 99186
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   year        99187 non-null  int64
 1   price       99187 non-null  int64
 2   mileage     99187 non-null  int64
```



```
 6   engineSize  99187 non-null  float64
dtypes: float64(2), int64(4), object(1)
memory usage: 5.3+ MB
```

```
df2["fuelType"].value_counts()
```

```
Petrol      54928
Diesel      40928
Hybrid       3078
Other         247
Electric        6
Name: fuelType, dtype: int64
```

```
f={"fuelType":{"Petrol":1,"Diesel":2,"Hybrid":3,"Other":4,"Electric":5}}
df2=df2.replace(f)
```

```
df2
```

	year	price	mileage	fuelType	tax	mpg	engineSize
0	2019	25000	13904	2	145	49.6	2.0
1	2019	26883	4562	2	145	49.6	2.0
2	2019	20000	7414	2	145	50.4	2.0
3	2019	33492	4825	1	145	32.5	2.0
4	2019	22900	6500	1	150	39.8	1.5
...	...	...	...	...	...	...	...
99182	2020	16999	4018	1	145	49.6	1.0
99183	2020	16999	1978	1	150	49.6	1.0
99184	2020	17199	609	1	150	49.6	1.0
99185	2017	19499	8646	1	150	47.9	1.4
99186	2016	15999	11855	1	150	47.9	1.4

99187 rows × 7 columns

```
y=df2["fuelType"]
x=df2.drop(["fuelType"],axis=1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

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

```
      RandomForestClassifier()
```

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

```
      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')
```

```
grid_search.best_score_
```

```
      0.9188535215324788
```

```
rfc_best=grid_search.best_estimator_
```

```
class_name=["Petrol","Diesel","Hybrid","Other","Electric"]
```

```
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],class_names=class_name,filled=True)
```

```
 Text(1339.2, 543.5999999999999, 'X[5] <= 0.5\ngini = 0.651\nsamples =
3818\nvalue = [5845, 90, 55, 11, 0]\nclass = Petrol'),
 Text(1264.8000000000002, 181.19999999999982, 'gini = 0.444\nsamples =
33\nvalue = [42, 21, 0, 0, 0]\nclass = Petrol'),
 Text(1413.6000000000001, 181.19999999999982, 'gini = 0.045\nsamples =
3785\nvalue = [5803, 69, 55, 11, 0]\nclass = Petrol'),
 Text(1636.8000000000002, 543.5999999999999, 'X[5] <= 1.45\ngini =
0.251\nsamples = 1375\nvalue = [1871, 323, 0, 0, 0]\nclass = Petrol'),
 Text(1562.4, 181.19999999999982, 'gini = 0.009\nsamples = 970\nvalue = [1527,
7, 0, 0, 0]\nclass = Petrol'),
 Text(1711.2, 181.19999999999982, 'gini = 0.499\nsamples = 405\nvalue = [344,
316, 0, 0, 0]\nclass = Petrol'),
 Text(2083.2000000000003, 906.0, 'X[3] <= 142.5\ngini = 0.466\nsamples =
12217\nvalue = [6379, 12502, 284, 46, 0]\nclass = Diesel'),
 Text(1934.4, 543.5999999999999, 'X[5] <= 2.4\ngini = 0.383\nsamples =
1026\nvalue = [148, 1258, 225, 7, 0]\nclass = Diesel'),
 Text(1860.0000000000002, 181.19999999999982, 'gini = 0.273\nsamples =
911\nvalue = [148, 1240, 72, 7, 0]\nclass = Diesel'),
 Text(2008.8000000000002, 181.19999999999982, 'gini = 0.188\nsamples =
115\nvalue = [0, 18, 153, 0, 0]\nclass = Hybrid'),
 Text(2232.0, 543.5999999999999, 'X[3] <= 327.5\ngini = 0.465\nsamples =
11191\nvalue = [6231, 11244, 59, 39, 0]\nclass = Diesel'),
 Text(2157.6000000000004, 181.19999999999982, 'gini = 0.463\nsamples =
11131\nvalue = [6138, 11234, 59, 39, 0]\nclass = Diesel'),
 Text(2306.4, 181.19999999999982, 'gini = 0.175\nsamples = 60\nvalue = [93, 10,
0, 0, 0]\nclass = Petrol'),
 Text(3515.4, 1630.8000000000002, 'X[4] <= 69.8\ngini = 0.519\nsamples =
12838\nvalue = [5494, 12738, 1846, 97, 2]\nclass = Diesel'),
 Text(2976.0, 1268.4, 'X[1] <= 12599.5\ngini = 0.486\nsamples = 9185\nvalue =
[5458, 8802, 141, 45, 1]\nclass = Diesel'),
 Text(2678.4, 906.0, 'X[2] <= 37016.0\ngini = 0.462\nsamples = 5107\nvalue =
[5204, 2823, 10, 35, 0]\nclass = Petrol'),
 Text(2529.6000000000004, 543.5999999999999, 'X[5] <= 1.45\ngini =
0.295\nsamples = 3469\nvalue = [4525, 949, 3, 29, 0]\nclass = Petrol'),
```

To undo cell deletion use Ctrl+M Z or the Undo option in the Edit menu

```
df3=pd.read_csv("C8_loan-train.csv")
df4=pd.read_csv("C8_loan-test.csv")
df3
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantInc
0	LP001002	Male	No	0	Graduate	No	5
1	LP001003	Male	Yes	1	Graduate	No	4
2	LP001005	Male	Yes	0	Graduate	Yes	3
3	LP001006	Male	Yes	0	Not Graduate	No	2
4	LP001008	Male	No	0	Graduate	No	6
...	...	...	...	...	...	...	
609	LP002978	Female	No	0	Graduate	No	2
610	LP002979	Male	Yes	3+	Graduate	No	4
611	LP002983	Male	Yes	1	Graduate	No	8
612	LP002984	Male	Yes	2	Graduate	No	7
613	LP002990	Female	No	0	Graduate	Yes	4

614 rows × 13 columns

```
df3["Loan_Status"]=df3["Loan_Status"].replace("Y",1,regex=True)
df3["Loan_Status"]=df3["Loan_Status"].replace("N",0,regex=True)
df3
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantInc
0	LP001002	Male	No	0	Graduate	No	5
1	LP001003	Male	Yes	1	Graduate	No	4
2	LP001005	Male	Yes	0	Graduate	Yes	3
3	LP001006	Male	Yes	0	Not Graduate	No	2
4	LP001008	Male	No	0	Graduate	No	6

```
df3_tr=df3.drop(["Dependents","Married","Loan_ID","Education","Gender","Property_Area","Loan_Status"],axis=1)
df3_tr
```

	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Te
0	No	5849	0.0	NaN	36(
1	No	4583	1508.0	128.0	36(
2	Yes	3000	0.0	66.0	36(
3	No	2583	2358.0	120.0	36(
4	No	6000	0.0	141.0	36(
...	...	...	...	...	
609	No	2900	0.0	71.0	36(
610	No	4106	0.0	40.0	18(
611	No	8072	240.0	253.0	36(
				187.0	36(
613	Yes	4583	0.0	133.0	36(

614 rows × 6 columns

To undo cell deletion use Ctrl+M Z or the Undo option in the Edit menu

```
df_tr=df3_tr.dropna()
```

```
df_tr.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 504 entries, 1 to 613
Data columns (total 6 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Self_Employed      504 non-null    object 
 1   ApplicantIncome    504 non-null    int64  
 2   CoapplicantIncome  504 non-null    float64
 3   LoanAmount         504 non-null    float64
 4   Loan_Amount_Term   504 non-null    float64
 5   Credit_History     504 non-null    float64
dtypes: float64(4), int64(1), object(1)
memory usage: 27.6+ KB
```

```
g1={"Self_Employed":{'Yes':1,'No':0}}
df_tr=df_tr.replace(g1).astype(int)
```

```
df_tr.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 504 entries, 1 to 613
Data columns (total 6 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Self_Employed      504 non-null    int32
 1   ApplicantIncome    504 non-null    int32
 2   CoapplicantIncome  504 non-null    int32
 3   LoanAmount         504 non-null    int32
 4   Loan_Amount_Term   504 non-null    int32
 5   Credit_History     504 non-null    int32
dtypes: int32(6)
memory usage: 15.8 KB
```

```
y=df_tr["Self_Employed"]
x=df_tr.drop(["Self_Employed"],axis=1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

```
df_tr["Self_Employed"].value_counts()
```

```
0    434
1     70
Name: Self_Employed, dtype: int64
```

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

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

```
RandomForestClassifier()
```

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

```
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')
```

```
grid_search.best_score_
```

```
0.8579545454545454
```

```
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],class_names=['Yes','No',"Yes"],filled=True)
```

To undo cell deletion use Ctrl+M Z or the Undo option in the Edit menu

```
 Text(1339.2, 543.5999999999999, 'X[5] <= 0.5\ngini = 0.651\nsamples =
3818\nvalue = [5845, 90, 55, 11, 0]\nclass = Yes'),
 Text(1264.8000000000002, 181.19999999999982, 'gini = 0.444\nsamples =
33\nvalue = [42, 21, 0, 0, 0]\nclass = Yes'),
 Text(1413.6000000000001, 181.19999999999982, 'gini = 0.045\nsamples =
3785\nvalue = [5803, 69, 55, 11, 0]\nclass = Yes'),
 Text(1636.8000000000002, 543.5999999999999, 'X[5] <= 1.45\ngini =
0.251\nsamples = 1375\nvalue = [1871, 323, 0, 0, 0]\nclass = Yes'),
 Text(1562.4, 181.19999999999982, 'gini = 0.009\nsamples = 970\nvalue = [1527,
7, 0, 0, 0]\nclass = Yes'),
 Text(1711.2, 181.19999999999982, 'gini = 0.499\nsamples = 405\nvalue = [344,
316, 0, 0, 0]\nclass = Yes'),
 Text(2083.2000000000003, 906.0, 'X[3] <= 142.5\ngini = 0.466\nsamples =
12217\nvalue = [6379, 12502, 284, 46, 0]\nclass = No'),
 Text(1934.4, 543.5999999999999, 'X[5] <= 2.4\ngini = 0.383\nsamples =
1026\nvalue = [148, 1258, 225, 7, 0]\nclass = No'),
 Text(1860.0000000000002, 181.19999999999982, 'gini = 0.273\nsamples =
911\nvalue = [148, 1240, 72, 7, 0]\nclass = No'),
 Text(2008.8000000000002, 181.19999999999982, 'gini = 0.188\nsamples =
115\nvalue = [0, 18, 153, 0, 0]\nclass = Yes'),
 Text(2232.0, 543.5999999999999, 'X[3] <= 327.5\ngini = 0.465\nsamples =
11191\nvalue = [6231, 11244, 59, 39, 0]\nclass = No'),
 Text(2157.6000000000004, 181.19999999999982, 'gini = 0.463\nsamples =
11131\nvalue = [6138, 11234, 59, 39, 0]\nclass = No'),
 Text(2306.4, 181.19999999999982, 'gini = 0.175\nsamples = 60\nvalue = [93, 10,
0, 0, 0]\nclass = Yes'),
 Text(3515.4, 1630.8000000000002, 'X[4] <= 69.8\ngini = 0.519\nsamples =
12838\nvalue = [5494, 12738, 1846, 97, 2]\nclass = No'),
 Text(2976.0, 1268.4, 'X[1] <= 12599.5\ngini = 0.486\nsamples = 9185\nvalue =
[5458, 8802, 141, 45, 1]\nclass = No'),
 Text(2678.4, 906.0, 'X[2] <= 37016.0\ngini = 0.462\nsamples = 5107\nvalue =
[5204, 2823, 10, 35, 0]\nclass = Yes'),
 Text(2529.6000000000004, 543.5999999999999, 'X[5] <= 1.45\ngini =
0.295\nsamples = 3469\nvalue = [4525, 949, 3, 29, 0]\nclass = Yes'),
                                                                  8\nsamples =

                                                                  613\nvalue = [0,
937, 3, 0, 0]\nclass = No'),
 Text(2827.2000000000003, 543.5999999999999, 'X[3] <= 5.0\ngini =
0.397\nsamples = 1638\nvalue = [679, 1874, 7, 6, 0]\nclass = No'),
 Text(2752.8, 181.19999999999982, 'gini = 0.257\nsamples = 237\nvalue = [310,
48, 3, 3, 0]\nclass = Yes'),
```

To undo cell deletion use Ctrl+M Z or the Undo option in the Edit menu

C9

```
df5=pd.read_csv("C9_Data.csv")
df5
```

	row_id	user_id	timestamp	gate_id
0	0	18	2022-07-29 09:08:54	7
1	1	18	2022-07-29 09:09:54	9
2	2	18	2022-07-29 09:09:54	9
3	3	18	2022-07-29 09:10:06	5
4	4	18	2022-07-29 09:10:08	5
...	...	...	...	...
37513	37513	6	2022-12-31 20:38:56	11
37514	37514	6	2022-12-31 20:39:22	6
37515	37515	6	2022-12-31 20:39:23	6
37516	37516	6	2022-12-31 20:39:31	9
37517	37517	6	2022-12-31 20:39:31	9

37518 rows × 4 columns

```
df5=df5.drop(["timestamp"],axis=1)
df5.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 37518 entries, 0 to 37517
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   row_id   37518 non-null  int64
 1   user_id  37518 non-null  int64
 2   gate_id  37518 non-null  int64
dtypes: int64(3)
memory usage: 879.5 KB
```

```
y=df5["user_id"]
x=df5.drop(["user_id"],axis=1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)


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

```
RandomForestClassifier()
```

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

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:666: UserWarning: The least populated class in y has o
  warnings.warn(("The least populated class in y has only %d"
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')
```

```
grid_search.best_score_
```

```
0.1067702383672226
```

```
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],class_names=['Yes','No','Yes'],filled=True)
```

To undo cell deletion use Ctrl+M Z or the Undo option in the Edit menu

C10

```
import pandas as pd
from sklearn.model_selection import train_test_split


from sklearn.ensemble import RandomForestClassifier
import matplotlib.pyplot as plt
from sklearn.model_selection import GridSearchCV
from sklearn.tree import plot_tree


df=pd.read_csv("C10_loan1.csv")
df
```

	Home Owner	Marital Status	Annual Income	Defaulted Borrower
0	Yes	Single	125	No
1	No	Married	100	No
2	No	Single	70	No
3	Yes	Married	120	No
4	No	Divorced	95	Yes
5	No	Married	60	No
6	Yes	Divorced	220	No
7	No	Single	85	Yes
8	No	Married	75	No
9	No	Single	90	Yes

```
df["Home Owner"]=df["Home Owner"].replace({"Yes":1,"No":0}).astype(int)
df["Marital Status"]=df["Marital Status"].replace({"Single":1,"Married":2,"Divorced":3}).astype(int)
df["Defaulted Borrower"]=df["Defaulted Borrower"].replace({"Yes":1,"No":0}).astype(int)
df
```

	Home Owner	Marital Status	Annual Income	Defaulted Borrower
0	1	1	125	0
1	0	2	100	0
2	0	1	70	0
3	1	2	120	0
4	0	3	95	1
5	0	2	60	0
6	1	3	220	0
7	0	1	85	1
8	0	2	75	0
9	0	1	90	1

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 4 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Home Owner          10 non-null     int32
 1   Marital Status      10 non-null     int32
 2   Annual Income       10 non-null     int64
 3   Defaulted Borrower  10 non-null     int32
dtypes: int32(3), int64(1)
memory usage: 328.0 bytes
```

```
y=df["Defaulted Borrower"]
x=df.drop(["Defaulted Borrower"],axis=1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

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

```
RandomForestClassifier()
```

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

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

```
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')
```

```
grid_search.best_score_
```

```
0.5833333333333333
```

```
rfc_best=grid_search.best_estimator_
```

```
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],class_names=['Yes','No'],filled=True)
```

```
[Text(2232.0, 1087.2, 'gini = 0.408\nsamples = 5\nvalue = [2, 5]\nclass = No')]
```