In [80]:
```
import pandas as pd
import numpy as np
d=pd.read_csv("eighthr.csv")
```

In [81]:
```
d.head()
```

Out[81]:

	1/1/1998	0.8	1.8	2.4	2.1	2	2.1.1	1.5	1.7	1.9	...	0.15	10.67	-1.56	5795	-12.1
0	1/2/1998	2.8	3.2	3.3	2.7	3.3	3.2	2.9	2.8	3.1	...	0.48	8.39	3.84	5805	14.05
1	1/3/1998	2.9	2.8	2.6	2.1	2.2	2.5	2.5	2.7	2.2	...	0.6	6.94	9.8	5790	17.9
2	1/4/1998	4.7	3.8	3.7	3.8	2.9	3.1	2.8	2.5	2.4	...	0.49	8.73	10.54	5775	31.15
3	1/5/1998	2.6	2.1	1.6	1.4	0.9	1.5	1.2	1.4	1.3	...	?	?	?	?	?
4	1/6/1998	3.1	3.5	3.3	2.5	1.6	1.7	1.6	1.6	2.3	...	0.09	11.98	11.28	5770	27.95

5 rows × 74 columns

In [82]:
```
d.isnull().sum()
```

Out[82]:
```
1/1/1998    0
0.8         0
1.8         0
2.4         0
2.1         0
           ..
17.9        0
10330       0
-55         0
0           0
0.          0
Length: 74, dtype: int64
```

In [83]:
```
cs=d.shape[1]
print(cs)
```

```
74
```

In [84]:
```
cr=d.shape[0]
print(cr)
```

```
2533
```

In [85]:
```
from sklearn.impute import SimpleImputer
imp_mean = SimpleImputer(missing_values='?', strategy='constant',fill_valu
d=imp_mean.fit_transform(d)
d=pd.DataFrame(d)
d.head()
```

Out[85]:

	0	1	2	3	4	5	6	7	8	9	...	64	65	66	67	68	
0	1/2/1998	2.8	3.2	3.3	2.7	3.3	3.2	2.9	2.8	3.1	...	0.48	8.39	3.84	5805	14.05	
1	1/3/1998	2.9	2.8	2.6	2.1	2.2	2.5	2.5	2.7	2.2	...	0.6	6.94	9.8	5790	17.9	
2	1/4/1998	4.7	3.8	3.7	3.8	2.9	3.1	2.8	2.5	2.4	...	0.49	8.73	10.54	5775	31.15	
3	1/5/1998	2.6	2.1	1.6	1.4	0.9	1.5	1.2	1.4	1.3	...	0.0	0.0	0.0	0.0	0.0	
4	1/6/1998	3.1	3.5	3.3	2.5	1.6	1.7	1.6	1.6	2.3	...	0.09	11.98	11.28	5770	27.95	4

5 rows × 74 columns

In [86]:
```
x=d.iloc[:,1:73]
y=d.iloc[:,[73]].astype('int')
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
dt=DecisionTreeClassifier()
l=LogisticRegression()
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
l=l.fit(x_train,y_train)
dt=dt.fit(x_train,y_train)
#y_pred=l.predict(x_test)
```

```
C:\Users\virupaksha\anaconda3\lib\site-packages\sklearn\utils\validation.
py:993: DataConversionWarning: A column-vector y was passed when a 1d arr
ay was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\virupaksha\anaconda3\lib\site-packages\sklearn\linear_model\_log
istic.py:814: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown i
n:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://s
cikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-re
gression (https://scikit-learn.org/stable/modules/linear_model.html#logis
tic-regression)
  n_iter_i = _check_optimize_result(
```

In [87]:
```
y_pred=l.predict(x_test)
from sklearn.metrics import confusion_matrix,roc_auc_score
matrix = confusion_matrix(y_test, y_pred)
# Accuracy
#from sklearn.metrics import accuracy_score
acc = (matrix[0,0]+matrix[1,1])/(matrix[0,0]+matrix[0,1]+matrix[1,0]+matri
# Recall
#from sklearn.metrics import recall_score
#rc=recall_score(y_test, y_pred)
# Precision
#from sklearn.metrics import precision_score
#pc=precision_score(y_test, y_pred)
p = matrix[0,0]/(matrix[0,0]+matrix[1,0])
r = matrix[0,0]/(matrix[0,0]+matrix[1,1])
f = matrix[0,0]/(matrix[0,0]+0.5*(matrix[1,0]+matrix[1,1]))
roc=roc_auc_score(y_test, y_pred)
print(acc)
print(p)
print(r)
print(f)
print(roc)
```

```
0.9368836291913215
0.9517102615694165
0.9957894736842106
0.9732510288065843
0.5301455301455301
```

In [88]:
```
y_pred=dt.predict(x_test)
matrix = confusion_matrix(y_test, y_pred)
# Accuracy
#from sklearn.metrics import accuracy_score
acc = (matrix[0,0]+matrix[1,1])/(matrix[0,0]+matrix[0,1]+matrix[1,0]+matri
# Recall
#from sklearn.metrics import recall_score
p = matrix[0,0]/(matrix[0,0]+matrix[1,0])
r = matrix[0,0]/(matrix[0,0]+matrix[1,1])
f = matrix[0,0]/(matrix[0,0]+0.5*(matrix[1,0]+matrix[1,1]))
roc=roc_auc_score(y_test, y_pred)
print(acc)
print(p)
print(r)
print(f)
print(roc)
```

```
0.9151873767258383
0.95625
0.9892241379310345
0.972457627118644
0.5732848232848233
```

In [ ]:
```
```

In [ ]: