In [1]:

```
import numpy as np
import pandas as pd
```

In [2]:

```
data=pd.read_csv(r"D:\ML_Course\Works_on_python\Decision tree & Random Forest Calssification\diabetes.csv")
```

In [3]:

```
data.head()
```

Out[3]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Diabetes
0	6	148	72	35	0	33.6	0.627	50	pos
1	1	85	66	29	0	26.6	0.351	31	neg
2	8	183	64	0	0	23.3	0.672	32	pos
3	1	89	66	23	94	28.1	0.167	21	neg
4	0	137	40	35	168	43.1	2.288	33	pos

In [4]:

```
data.isnull().any()
```

Out[4]:

```
Pregnancies                 False
Glucose                     False
BloodPressure               False
SkinThickness               False
Insulin                     False
BMI                         False
DiabetesPedigreeFunction    False
Age                         False
Diabetes                    False
dtype: bool
```

In [5]:

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
data["Diabetes"]=le.fit_transform(data["Diabetes"])
```

In [6]:
```
data.head()
```

Out[6]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Diabetes
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

In [7]:
```
x=data.iloc[:,0:8].values
y=data.iloc[:,8:9].values
```

In [8]:
```
x.shape
```

Out[8]:
```
(768, 8)
```

In [9]:
```
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,random_state=0)
```

In [10]:
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
x_train=sc.fit_transform(x_train)
x_test=sc.fit_transform(x_test)
```

DecisionTreeClassifier

```
In [11]: from sklearn.tree import DecisionTreeClassifier
         dtc=DecisionTreeClassifier(criterion="entropy",random_state=0)
         dtc.fit(x_train,y_train)

Out[11]: DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=None,
                     max_features=None, max_leaf_nodes=None,
                     min_impurity_decrease=0.0, min_impurity_split=None,
                     min_samples_leaf=1, min_samples_split=2,
                     min_weight_fraction_leaf=0.0, presort=False, random_state=0,
                     splitter='best')

In [12]: y_predict=dtc.predict(x_test)

In [13]: y_predict

Out[13]: array([1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0,
                0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1,
                1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1,
                1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0,
                1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1,
                0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0,
                0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0])

In [14]: from sklearn.metrics import accuracy_score
         accuracy_score(y_test,y_predict)

Out[14]: 0.7012987012987013

In [15]: from sklearn.metrics import confusion_matrix
         cm=confusion_matrix(y_test,y_predict)

In [16]: cm

Out[16]: array([[78, 29],
                [17, 30]], dtype=int64)

In [17]: import sklearn.metrics as metrics
         fpr,tpr,threshold=metrics.roc_curve(y_test,y_predict)
         roc_auc=metrics.auc(fpr,tpr)
```

In [18]:

```
import matplotlib.pyplot as plt
plt.title("roc")
plt.plot(fpr,tpr,'b',label = 'auc = %0.2f'%roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0,1],[0,1],'r--')
plt.xlim([0,1])
plt.ylim([0,1])
plt.ylabel('tpr')
plt.xlabel('fpr')
```

Out[18]:

```
Text(0.5, 0, 'fpr')
```

In [19]:

```
dtc.predict(sc.transform([[1,150,80,24,33,46,4,66]]))
```

Out[19]:

```
array([1])
```

In [20]:

```
data.head(1)
```

Out[20]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Diabetes
0	6	148	72	35	0	33.6	0.627	50	1

RandomForestClassifier

In [21]:

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators = 1000,criterion = 'entropy',random_state = 0)
```

In [22]:

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

```
C:\Users\anikp\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: DataConversionWarning: A column-vector y was passed when a
1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  """Entry point for launching an IPython kernel.
```

Out[22]:

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='entropy',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=1000, n_jobs=None,
            oob_score=False, random_state=0, verbose=0, warm_start=False)
```

In [23]:
```
y_pred1 = rfc.predict(x_test)
```

In [24]:
```
y_pred1
```

Out[24]:
```
array([1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1,
       1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1,
       1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1,
       0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0])
```

In [25]:
```
y_test
```

Out[25]:
```
array([[1],
       [0],
       [0],
       [1],
       [0],
       [0],
       [1],
       [1],
       [0],
       [0],
       [1],
       [1],
       [0],
       [0],
       [0],
       [0],
       [1],
       [0],
       [0],
```

In [26]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_pred1,y_test)
```

Out[26]:
```
0.8181818181818182
```

In [27]:
```
from sklearn.metrics import confusion_matrix
cm1 = confusion_matrix(y_test,y_pred1)
```

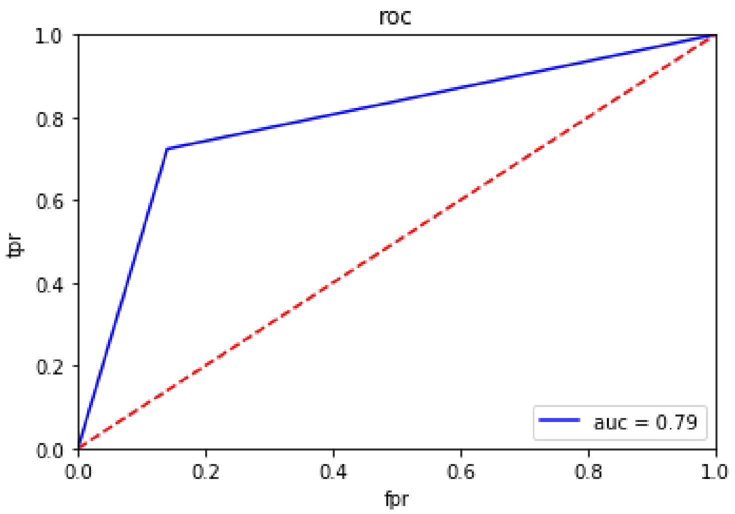
```
In [28]: cm1
```

```
Out[28]: array([[92, 15],
                [13, 34]], dtype=int64)
```

```
In [29]: import sklearn.metrics as metrics
         fpr1,tpr1 , threshold = metrics.roc_curve(y_test,y_pred1)
         roc_auc1 = metrics.auc(fpr1,tpr1)
```

```
In [30]: plt.title("roc")
         plt.plot(fpr1,tpr1,'b',label = 'auc = %0.2f'%roc_auc1)
         plt.legend(loc = 'lower right')
         plt.plot([0,1],[0,1],'r--')
         plt.xlim([0,1])
         plt.ylim([0,1])
         plt.ylabel('tpr')
         plt.xlabel('fpr')
```

```
Out[30]: Text(0.5, 0, 'fpr')
```



Logistic Regresion

In [31]:

```
from sklearn.linear_model import LogisticRegression
log=LogisticRegression()
log.fit(x_train,y_train)
```

```
C:\Users\anikp\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: Default solver will be changed
to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\anikp\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning: A column-vector y was passe
d when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[31]:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='warn',
          tol=0.0001, verbose=0, warm_start=False)
```

In [32]:

```
y_pred2=log.predict(x_test)
```

In [33]:

```
cm2=confusion_matrix(y_test,y_pred2)
```

In [34]:

```
cm2
```

Out[34]:

```
array([[94, 13],
       [18, 29]], dtype=int64)
```

In [35]:

```
accuracy_score(y_test,y_pred2)
```

Out[35]:

```
0.7987012987012987
```

In [36]:

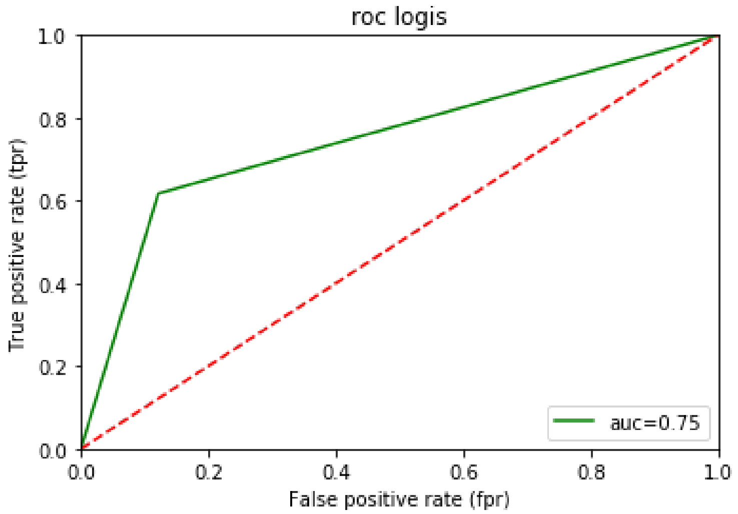
```
fpr3,tpr3,threshold3=metrics.roc_curve(y_test,y_pred2)
roc_auc3=metrics.auc(fpr3,tpr3)
```

In [37]:

```
import matplotlib.pylab as plt
plt.title("roc logis")
plt.plot(fpr3,tpr3,'g',label='auc=%0.2f'%roc_auc3)
plt.legend(loc = 'lower right')
plt.plot([0,1],[0,1],'r--')
plt.xlim([0,1])
plt.ylim([0,1])
plt.ylabel('True positive rate (tpr)')
plt.xlabel('False positive rate (fpr)')
```

Out[37]:

```
Text(0.5, 0, 'False positive rate (fpr)')
```



K Nearest Neighbor

In [38]:
```
from sklearn.neighbors import KNeighborsClassifier
knn=KNeighborsClassifier(n_neighbors=5,metric='minkowski')
knn.fit(x_train,y_train)
```

```
C:\Users\anikp\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: DataConversionWarning: A column-vector y was passed when a
1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  This is separate from the ipykernel package so we can avoid doing imports until
```

Out[38]:
```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                     weights='uniform')
```

In [39]:
```
ypred3=knn.predict(x_test)
```

In [40]:
```
cm3=confusion_matrix(y_test,ypred3)
```

In [41]:
```
cm3
```

Out[41]:
```
array([[96, 11],
       [17, 30]], dtype=int64)
```

In [42]:
```
accuracy_score(y_test,ypred3)
```

Out[42]:
```
0.8181818181818182
```

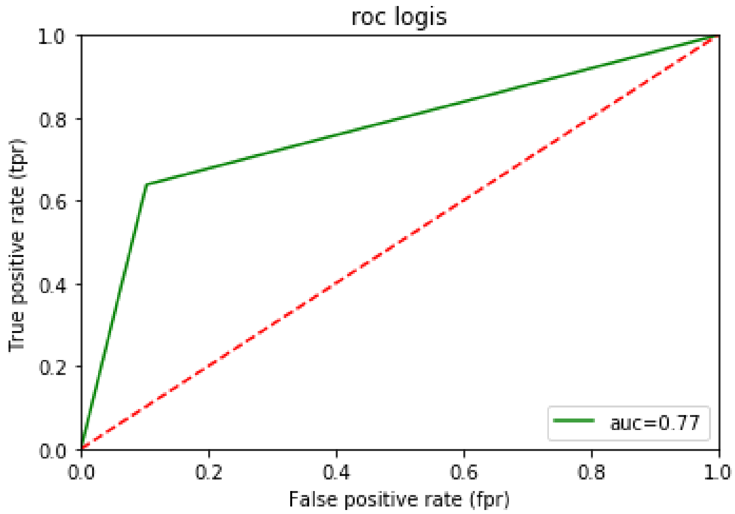
In [43]:
```
fpr4,tpr4,threshold4=metrics.roc_curve(y_test,ypred3)
roc_auc4=metrics.auc(fpr4,tpr4)
```

In [44]:

```
import matplotlib.pylab as plt
plt.title("roc logis")
plt.plot(fpr4,tpr4,'g',label='auc=%0.2f'%roc_auc4)
plt.legend(loc = 'lower right')
plt.plot([0,1],[0,1],'r--')
plt.xlim([0,1])
plt.ylim([0,1])
plt.ylabel('True positive rate (tpr)')
plt.xlabel('False positive rate (fpr)')
```

Out[44]:

```
Text(0.5, 0, 'False positive rate (fpr)')
```



In [ ]: