```
import numpy as np
import random
import statistics as stats
import math
import pandas as pd
import matplotlib.pyplot as plt
import sklearn
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegressionCV
from sklearn import preprocessing
from sklearn import tree
from sklearn.metrics import accuracy_score
from sklearn import ensemble

x=pd.read_csv('heartdata.csv')
y=pd.read_csv('heartlabel.csv')
x=pd.DataFrame(x)

y=pd.DataFrame(y)
leni,lenj=x.shape
x=np.mat(x)
y=np.mat(y)

yzero,yone=0,0
for i in range(len(y)):
    if y[i]==[0]:
        yzero+=1
    else:
        yone+=1
print(yzero,yone)

x=x.T

x1=x[4]
x2=x[7]
x1=pd.DataFrame(x1)
x2=pd.DataFrame(x2)

x1list=[]
x2list=[]
x1flist=[]
x2flist=[]
print(len(y))

for i in range(len(y)):
```

```
    if y[i]==[0]:
        x1list.append(x1.iloc[0,i])
        x2list.append(x2.iloc[0,i])
    else:
        x1flist.append(x1[i])
        x2flist.append(x2[i])

plt.scatter(x1list,x2list,color='red')
plt.scatter(x1flist,x2flist,color='blue')
plt.show()



for j in range(lenj):
    x[j]-=np.mean(x[j])
    x[j]=x[j]/np.linalg.norm(x[j])
x=x.T
X_train, X_test, Y_train, Y_test = train_test_split(x, y,
test_size=0.2, random_state=42)
X_train,X_val,Y_train,Y_val=train_test_split(X_train,Y_train,tes
t_size=0.3,random_state=42)




#firstly use logistic regression,pre for train is only 47%

for Cnum in [0.01,0.1,0.5,1,2,5,10,100]:
    for solver in ['liblinear','saga']:

model=sklearn.linear_model.LogisticRegression(C=Cnum,penalty='l1
',solver=solver)
        model.fit(X_train,Y_train)
        y_pred=model.predict(X_val)
        result=accuracy_score(Y_val,y_pred)
        print(Cnum,solver,result)
for Cnum in [0.01,0.1,0.5,1,2,5,10,100]:
    for solver in ['lbfgs','sag']:

model=sklearn.linear_model.LogisticRegression(C=Cnum,penalty='l2
',solver=solver)
        model.fit(X_train,Y_train)
        y_pred=model.predict(X_test)
        result=accuracy_score(Y_test,y_pred)
```

```
model=sklearn.linear_model.LogisticRegression(C=10,penalty='l1',
solver='liblinear')
model.fit(X_train,Y_train)
y_pred=model.predict(X_val)
result=accuracy_score(Y_val,y_pred)
print('val is',result)
y_pred=model.predict(X_test)
result2=accuracy_score(Y_test,y_pred)
print('test is',result2)

#the second way is tree and forest'''
for length in [6,7,8,9,10,11]:
    for nodes in [20,30,40,50,60,70]:
        clf=tree.DecisionTreeClassifier()
        clf.fit(X_train,Y_train)
        y_pred = clf.predict(X_val)
        result = accuracy_score(Y_val, y_pred)
        print('val is',length,nodes,result)

        y_pred=clf.predict(X_test)
        result2=accuracy_score(Y_test,y_pred)
        print('test is',length,nodes,result2)

        plt.figure(figsize=[32,24])

tree.plot_tree(clf.fit(X_train,Y_train),filled=True,label='all',
impurity=True,)
        plt.show()


#random forest the result is totally same with decision trees.

clf=sklearn.ensemble.RandomForestClassifier()
clf.fit(X_train,Y_train)
y_pred = clf.predict(X_val)
result = accuracy_score(Y_val, y_pred)
print('val is',result)
y_pred=clf.predict(X_test)
result2=accuracy_score(Y_test,y_pred)
print('test is',result2)
```