决策树  
import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
import sklearn  
  
df=pd.read\_csv(r"C:\Users\阿韩想养二哈\Desktop\Titanic数据集\Titanic数据集\train.csv")  
#舍去无关变量  
df=df.drop(['PassengerId','Cabin','Name','Ticket','SibSp','Parch','Fare'],axis=1)  
print(df.head())  
#查看是否有缺失值  
print(df.info())  
# 数据缺失值个数  
total = df.isnull().sum().sort\_values(ascending=False)  
# 缺失值数据比例  
percent = (df.isnull().sum() / df.isnull().count()).sort\_values(ascending=False)  
print(total)  
print(percent)  
# 缺失值填充  
df['Age'] = df['Age'].fillna(0)#用0填充  
print(df.isnull().sum().max())  
  
print(df.head(5))#查看数据集前5行数据  
print(df.describe())  
#数据类型分类float64(2), int64(5), object(5)  
  
#查看是否有重复值  
print(df[df.duplicated()])#此时返回一个[]，无重复值  
#查看每列数值是否唯一  
all\_cloumns=list(df)  
print(all\_cloumns)#以列表形式返回所有列名  
print(df.nunique())#粗略查看每列唯一值数量  
for i in all\_cloumns[0:96]:  
 print(df[i].nunique())#详细查看每列唯一值的数量，发现id列只是个计数列，target总共有九种  
# 未添加处理重复值的代码  
  
# 数据可视化  
# 首先查看性别的分布情况  
diction = dict(df['Sex'].value\_counts()) # 统计不同分类结果的数量  
print(diction)  
  
values = diction.values()  
label = diction.keys()  
plt.pie(values, labels=label, autopct='%10.1f%%') # 绘制饼图  
plt.show()  
# plt.savefig('C:/Users/27812/Desktop/a.png', dpi=300)#保存图片  
  
# 将不符合机器学习要求的数据进行哑变量处理  
#Pclass,Sex为数值型变量，将其转化为类别型，进行哑变量处理  
df.Pclass=df.Pclass.astype('category')  
#哑变量处理  
dummy=pd.get\_dummies(df[['Sex','Pclass','Embarked']])  
#水平合并原数据集和哑变量  
df=pd.concat([df,dummy],axis=1)  
#删除为处理时时变量，保留哑变量  
df.drop(['Sex','Pclass','Embarked'],inplace=True,axis=1)  
print(df.head())  
#df.to\_excel('Titanic.xlsx')  
  
# 这里采用seaborn库来可视化  
# 抽取部分特征进行可视化  
df\_1 = df.loc[200:400, :] # 选取前4个特征值对应的200行数据进行可视化  
  
#sns.pairplot函数用于创建一个散点图矩阵  
pairplot\_fig1 = sns.pairplot(df\_1, kind='reg',#kind='reg'表示用线性回归模型拟合每个变量的对  
 diag\_kind='hist',#diag\_kind='hist'表示在散点图的每个诊断图中绘制直方图  
 hue='Survived',#hue='target'表示根据目标变量（'target'）对数据进行颜色区分  
 vars=['Sex\_female', 'Age', 'Embarked\_C'])  
#fig\_path1 = 'C:/Users/27812/Desktop/图片1'  
#pairplot\_fig1.savefig(fig\_path1, dpi=500) # 保存图片  
# 选取前4个特征对应的所有行数据进行可视化  
pairplot\_fig2 = sns.pairplot(df, kind='reg',  
 diag\_kind='hist',  
 hue='Survived',  
 vars=['Sex\_female', 'Age', 'Embarked\_C'])  
#fig\_path1 = 'C:/Users/27812/Desktop/图片2'  
#pairplot\_fig2.savefig(fig\_path1, dpi=500) # 保存图片  
  
# sns.jointplot函数是一个更复杂的联合分布图  
jointplot\_fig3 = sns.jointplot(x='Age', y='Survived', data=df, kind='kde')  
#fig\_path2 = 'C:/Users/27812/Desktop/图片3'  
#jointplot\_fig3.savefig(fig\_path2, dpi=500) # 保存图片  
  
'''  
#注：决策树不是基于距离计算的算法模型，所以不需要归一化  
'''  
  
# 划分数据集  
from sklearn.model\_selection import train\_test\_split#用于将数据集划分为训练集和测试集  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(df.drop(['Survived'], axis=1),#选取了数据集df的第2到第95列作为特征数据  
 df["Survived"],#选取了数据集df的最后一列作为目标变量  
 test\_size=0.3)#将30%的数据作为测试集  
  
#导入  
from sklearn import tree  
#实例化  
clf = tree.DecisionTreeClassifier(criterion='gini',#选择基尼系数 基尼系数(Gini impurity)是决策树中的一个重要参数，用于衡量纯度,基尼系数越小，纯度越高。  
 random\_state=42,#随机种子(seed)，用于保证每次运行程序时得到的结果都是一样的  
 splitter='random',#在决策树的每个节点处，是随机选择一个特征来进行切分  
 max\_depth=6,#决策树的最大深度为30  
 min\_samples\_leaf=10,#每个叶子节点最少需要有10个样本  
 min\_samples\_split=2) #决定是否将一个节点进行切分时，至少需要10个样本  
#使用训练数据x\_train和y\_train来拟合（fit）这个决策树分类器  
clf = clf.fit(x\_train,y\_train) #训练  
#测试数据x\_test和y\_test来评估这个决策树分类器的性能，返回的是分类器的得分  
result = clf.score(x\_test,y\_test) #为测试集打分  
print('测试集打分',result)  
print('训练集打分',clf.score(x\_train,y\_train))