##加载工具包pandas

import pandas as pd

##读取数据，文件为csv格式，需要修改路径

housing = pd.read\_csv("D:/E drive/SZTU/introduction of machine learning/handson-ml-master/datasets/housing/housing.csv")

##设置下面这些方程显示的格式，为全部列都显示，如果不加这行，就会省略一些列。可以自己试一下

pd.set\_option('display.max\_columns', None)

##看一下前五行的数据是怎样的

housing.head()

##快速获取数据集的简单描述，特别是总行数、每个属性的类型和非空值的数量

housing.info()

##因为ocean\_proximity为分类属性，使用value\_counts计算和查看每类别的频数

housing["ocean\_proximity"].value\_counts()

##显示数值属性的摘要，包括最小值、最大值、标准差、均值等

housing.describe()

##加载工具包matplotlib

import matplotlib.pyplot as plt

##在整个数据集上绘制直方图

housing.hist(bins=50, figsize=(20,15))

plt.show()

##设置一个随机数生成器的种子，从而让它始终生成相同的随机索引

np.random.seed(42)

##加载工具包numpy

import numpy as np

## 手动写方程去分训练集和测试集，其中data为数据集，test\_ratio为测试集占数据集大小的比率，通常为20%，如果数据集很大，比率将更小

def split\_train\_test(data, test\_ratio):

shuffled\_indices = np.random.permutation(len(data))

test\_set\_size = int(len(data) \* test\_ratio)

test\_indices = shuffled\_indices[:test\_set\_size]

train\_indices = shuffled\_indices[test\_set\_size:]

return data.iloc[train\_indices], data.iloc[test\_indices]

##把房地产数据（housing）用刚定义的方程split\_train\_test随机分为训练集和测试集，测试集大学为数据的20%

train\_set, test\_set = split\_train\_test(housing, 0.2)

print(len(train\_set), "train +", len(test\_set), "test") #打印出训练集和测试集的大小

##从sklearn包里加载train\_test\_split函数，用于直接分训练集和测试集

from sklearn.model\_selection import train\_test\_split

##使用train\_test\_split函数将房地产数据分为80%的训练集和20%的测试集，其中random\_state参数为随机生成器种子

train\_set, test\_set = train\_test\_split(housing, test\_size=0.2, random\_state=42)

##查看测试集数据的前五行

test\_set.head()

##对于收入中位数画一个直方图，因为要预测房价中位数，收入中位数是一个非常重要的

属性，要按照收入中位数进行分层抽样

housing["median\_income"].hist()

##创建5个收入类别属性的（用1～5来做标签），0～1.5是类别1，1.5～3是类别2，得到的结果是在房地产housing数据多了一列，这一列表示每一行的数据属于哪一个收入类别

housing["income\_cat"] = pd.cut(housing["median\_income"],

bins=[0., 1.5, 3.0, 4.5, 6., np.inf],

labels=[1, 2, 3, 4, 5])

##对收入类别做一个计数，统计每一个类别的频数

housing["income\_cat"].value\_counts()

##对新建的这个收入类别画一个直方图，图中有五个类，高度代表频数

housing["income\_cat"].hist()

##从sklearn包里面加载StratifiedShuffleSplit函数，用来做分层抽样

from sklearn.model\_selection import StratifiedShuffleSplit

##根据收入类别进行分层抽样，参数 n\_splits是将训练数据分成训练集/测试集对的组数，参数test\_size是用来设置测试集所占的比例。分好的训练师数集为strat\_train\_set，测试数集为strat\_test\_set

##关于StratifiedShuffleSplit函数详见https://blog.csdn.net/wangxuecheng666/article/details/109246126

##关于.loc和.iloc的详细内容参https://blog.csdn.net/sushangchun/article/details/83514803

split = StratifiedShuffleSplit(n\_splits=1, test\_size=0.2, random\_state=42)

for train\_index, test\_index in split.split(housing, housing["income\_cat"]):

strat\_train\_set = housing.loc[train\_index]

strat\_test\_set = housing.loc[test\_index]

##查看测试数集中收入中位数各类别所占的比例

strat\_test\_set["income\_cat"].value\_counts() / len(strat\_test\_set)

##查看全部房地产housing数集中收入中位数各类别所占的比例，发现与之前的测试集一样

housing["income\_cat"].value\_counts() / len(housing)

##对于训练集和测试集数据，删掉收入类别属性，恢复原来的数据样式，关于pandas的drop函数的参数设置，详细见https://blog.csdn.net/nuaadot/article/details/78304642

for set\_ in (strat\_train\_set, strat\_test\_set):

set\_.drop("income\_cat", axis=1, inplace=True)

##将训练集直接copy为housing，注意这里housing替代了原来的数据集，当然你也可以用别的名字来命名，保留原始的数据集

housing = strat\_train\_set.copy()

##对数据画散点图，每一行数据代表一个点，横坐标是经度，纵坐标是纬度

housing.plot(kind="scatter", x="longitude", y="latitude")

##对数据画散点图，将alpha选项（透明度）设置为0.1，可以更清楚地看出高密度数据点的位置

housing.plot(kind="scatter", x="longitude", y="latitude", alpha=0.1)

##每个圆的半径大小代表了每个区域的人口数量（选项s），颜色代表价格（选项c）。我们使用一个名叫jet的预定义颜色表（选项cmap）来进行可视化，颜色范围从蓝（低）到红（高）

housing.plot(kind="scatter", x="longitude", y="latitude", alpha=0.4,

s=housing["population"]/100, label="population", figsize=(10,7),

c="median\_house\_value", cmap=plt.get\_cmap("jet"), colorbar=True,

sharex=False)

##使用corr（）方法计算出训练数据housing的每对属性之间的标准相关系数

corr\_matrix = housing.corr()

##在上面计算的相关系数矩阵中，取出房价中位数这一列，考察所有变量与房价中位数的相关系数，并且从大到小排序

corr\_matrix["median\_house\_value"].sort\_values(ascending=False)

##从pandas包中加载scatter\_matrix函数，用来画每两个变量之间的散点图，绘制出每个数值属性相对于其他数值属性的相关值

from pandas.plotting import scatter\_matrix

##由于篇幅限制，只画与房价中位数相关度比较高的几个变量的图，注意对角线为变量的直方图

attributes = ["median\_house\_value", "median\_income", "total\_rooms",

"housing\_median\_age"]

scatter\_matrix(housing[attributes], figsize=(12, 8))

##画收入中位数与房价中位数的散点图，透明度alpha设为0.1

housing.plot(kind="scatter", x="median\_income", y="median\_house\_value",

alpha=0.1)

##设置x轴和y轴的数值范围

plt.axis([0, 16, 0, 550000])

##尝试不同属性的变换和组合，再来计算相关系数

housing["rooms\_per\_household"] = housing["total\_rooms"]/housing["households"]

housing["bedrooms\_per\_room"] = housing["total\_bedrooms"]/housing["total\_rooms"]

housing["population\_per\_household"]=housing["population"]/housing["households"]

##重新计算训练数据的相关系数矩阵，现在一共有12个属性

corr\_matrix = housing.corr()

##对不同变量与房价中位数的相关系数从高到低排序

corr\_matrix["median\_house\_value"].sort\_values(ascending=False)