燕山大学

Python 机器学习实验报告

Python Machine Learning Experiment Report

学生所在学院：信息科学与工程学院（软件学院）

学生所在班级：软件工程 2018 级 6 班

学生姓名：乔翱

学生学号：201811040809

指导教师：于浩洋 李可 郝晓冰

教 务 处

2021 年 2 月

目录

实验一 线性回归

一、实验目的

1.理解并掌握经典的线性回归模型。

2.熟悉并掌握 AI Studio 实践平台的账户创建与实践基本操作。

3.能够基于线性回归模型进行数据分析与预测。

二、实验内容

1.在 AI Studio 实践平台创建账户，加入本课程。

2.熟悉 AI Studio 实践平台的基本操作。

3.对于给定的 3 个例题,基于线性回归模型进行波士顿房价预测、疫情预测等练习。

4.对于给定的 2 个项目，自行编写程序，基于线性回归模型对糖尿病、影厅观影人数进行回归分析与预测。

三、实验过程

1、熟悉 AI Studio 平台

AI Studio 是百度推出的基于 PaddlePaddle 框架的一站式深度学习平台，百度提供 Jupyter notebook 的定制修改版本的编程环境，并且提供免费 GPU 算力加速模型开发，类似于谷歌 colab。百度推出的 AI Studio 是一个一站式开发平台：囊括了 AI 教程、代码环境、算法算力、数据集，并提供免费的在线云计算，是一个一体化编程环境。

之前并没有接触过这个平台，通过本次实验了解到这个平台，可以熟练使用这个平台的资源，通过 AI Studio 平台可以很方便对机器学习、深度学习等知识进行学习。

2、对线性回归的认识和理解

两个变量之间的关系是一次函数关系的，也就是图象是直线，叫做线性。人们在测量事物的时候因为客观条件所限，求得的都是测量值，而不是事物真实的值，为了能够得到真实值，无限次的进行测量，最后通过这些测量数据计算回归到真实值，这就是回归。通过对大量的观测数据进行处理，从而得到比较符合事物内部规律的数学表达式。也就是说寻找到数据与数据之间的规律所在，从而就可以模拟出结果，也就是对结果进行预测，通过已知的数据得到未知的结果。

3、对例题的理解

例题 1 是对波士顿房价的预测，是一个单变量线性回归的问题。对于这个例题，我认为在选择特征的时候，可以选择多个特征，进行多变量线性回归预测，选择特征的时

候可以选择与房价相关性最高的几个特征，例如选出和房价的相关系数的绝对值大于0.5的特征，经过测试，这样训练出的线性模型的预测准确率更高。

例题2主要对比了三类线性回归模型，包括线性回归、岭回归、套索回归。岭回归和套索回归使用了正则化项，也就是给损失函数加上一个参数项，利用参数项可以控制参数幅度，限制参数搜索空间，解决欠拟合和过拟合问题。岭回归使用了L2正则项，而套索回归使用了L1正则化，二者都可以很好地增强模型泛化能力，解决欠拟合和过拟合问题。

例题3基于线性回归模型预测疫情，我认为利用线性回归模型来对疫情进行预测，不是特别准备，因为疫情确诊人数的增加会受到很多因素的影响，可能某一天突然增多很多，也可能某一天突然减小很多，所以我觉得对于疫情的预测采用线性回归模型，预测的准确率不是很高。

4、糖尿病预测

（1）项目实现思路

本项目是针对糖尿病数据集进行分析，并预测病情的。首先先对糖尿病数据集进行数据分析，观察数据集，观察数据的特征及分布；之后划分数据集，把数据集划分为训练集和测试集；完后建立线性模型，训练数据，模型预测，评估模型。

分析每个特征和结果之间的关系，找出线性系数最大的几个特征，提取出线性相关性最大的几个特征，配置线性模型，训练数据，评估模型，最后对预测结果可视化。

（2）基本流程

1.载入糖尿病情数据库diabetes，查看数据。

2.切分数据，组合成DateFrame数据，并输出数据集前几行，观察数据。

3.查看数据集信息，从数据集中抽取训练集和测试集。

4.建立线性回归模型，训练数据，评估模型。

5.考察每个特征值与结果之间的关系，分别以散点图展示。

思考：根据散点图结果对比，哪个特征值与结果之间的相关性最高？

6.把5中相关性最高的特征值提取，然后进行数据切分。

7.创建线性回归模型，进行线性回归模型训练。

8.对测试集进行预测，求出权重系数。

9.对预测结果进行评价，结果可视化。

（3）核心代码

数据集的加载以及分析，划分数据集。

```
diabetes= load_diabetes()
```

```
data =  diabetes['data']
target = diabetes['target']
feature_names = diabetes['feature_names']
df = pd.DataFrame(data,columns = feature_names)
df.describe()
X_train,X_test,Y_train,Y_test=train_test_split(data,target,train_size
=0.75)
```

配置线性模型，利用线性模型对数据集进行分析。

```
model = LinearRegression()
model.fit(X_train,Y_train)
model.score(X_test,Y_test)
```

考察每个特征值与结果之间的关联性。

```
plt.figure(figsize=(2*6,5*5))
for i,col in enumerate(df.columns):
    train_X = df.loc[:,col]
    train_X=np.array(train_X)
    train_X=train_X.reshape(-1,1)
    train_Y = target
    le= LinearRegression()
    le.fit(train_X,train_Y)
    score = le.score(train_X,train_Y)
    axe = plt.subplot(5,2,i+1)
    plt.scatter(train_X,train_Y)
    k =  le.coef_
    b =  le.intercept_
    x = np.linspace(train_X.min(),train_X.max(),100)
    y = k * x + b
    plt.plot(x,y,c='red')
    axe.set_title(col + ':' + str(score))
plt.show()
```

选取了三个与结果相关性最大的变量，利用线性模型进行预测

```
df=df[['bmi','s5','diabetes']]
y=np.array(df['diabetes'])
df=df.drop(['diabetes'],axis=1)
X=np.array(df)
train_X,test_X,train_Y,test_Y=train_test_split(X,y,test_size=0.25)
model=LinearRegression()
model.fit(train_X,train_Y)
model.score(test_X,test_Y)
```

```
print ('求解截距项为：',model.intercept_)
print ('求解系数为：',model.coef_)
y_hat = model.predict(test_X) #对测试集的预测
```

对预测结果评价，结果可视化。

```
plt.figure(figsize=(10,6))
t=np.arange(len(test_X))
plt.plot(t,test_Y,'r',linewidth=2,label='y_test')
plt.plot(t,y_hat,'g',linewidth=2,label='y_hat')
plt.xlabel('test data')
plt.ylabel('diabetes')
```

（4）调试过程

本次实验调试，主要是特征的选取，在选取特征的时候，尝试只选取与结果相关性最高的特征、选取和结果相关性前二的特征、选取与结果相关性前三的特征，三种选取特征的方法，但是训练出来的模型预测准确率都大概只有百分之五十左右，我觉得其中的原因一个在于线性模型本身就是一个很简单的模型，模型的预测效果不是特别好，另外就是特征工程的问题，对于此数据集没有进一步深入研究，探索数据，做好特征工程。

5、影厅观影人数预测（多变量线性回归）

（1）项目实现思路

首先对数据集进行读取，完后绘制影厅观影人数和影厅面积的散点图，绘制散点图矩阵，划分数据集，建立线性回归模型训练，预测，评估，对预测结果可视化。

（2）基本流程

1. 读取给定文件中数据集文件。（数据集路径：data/data72160/1_film.csv）
2. 绘制影厅观影人数（filmnum）与影厅面积（filmsize）的散点图。
3. 绘制影厅人数数据集的散点图矩阵。
4. 选取特征变量与相应变量，并进行数据划分。
5. 进行线性回归模型训练。
6. 根据求出的参数对测试集进行预测。
7. 绘制测试集相应变量实际值与预测值的比较。
8. 对预测结果进行评价。

（3）核心代码

加载数据，分析数据。

```
data=pd.read_csv('data/data72160/1_film.csv')
data.head()
```

绘制 filenum 与 filesize 的散点图。

```
X = data.iloc[:,1:2]
y = data.iloc[:,0:1]
plt.scatter(X, y)
plt.xlabel('filesize')
plt.ylabel('filenum')
plt.title('The relation of filenum and filesie')
plt.show()
```

绘制散点图矩阵。

```
X = data.iloc[:,1:4]
y = data.filmnum
X = np.array(X.values)
y = np.array(y.values)
df = pd.DataFrame(X,columns=['filesize','ratio','quality'])
grr = pd.plotting.scatter_matrix(df,c=y,figsize=(15,15),marker='o',his
t_kwds={'bins':20},s=60,alpha=.8)
plt.savefig('./test.jpg')
plt.show()
```

配置线性模型，训练模型，模型预测，评估模型，结果可视化。

```
ridge = linear_model.Ridge(alpha = 0.1)
ridge.fit(x_train,y_train)
y_pre = ridge.predict(x_test)
print(y_pre)
plt.figure(figsize=(10,6))
t = np.arange(0,len(x_test),1)
plt.plot(t,y_pre,'b',linewidth = 2,label = 'predict')
plt.plot(t,y_test,'r',linewidth = 2,label = 'test')
plt.legend()
plt.show()
```

（4）调试过程

本项目和之前的项目的处理过程类似，所以在编写程序时很顺利，没有经过很多调试，对于散点图矩阵，由于之前并没有接触过，所以上网查询了相关资料，成功画出散点图矩阵。

四、实验结果

1、特征与结果的相关性

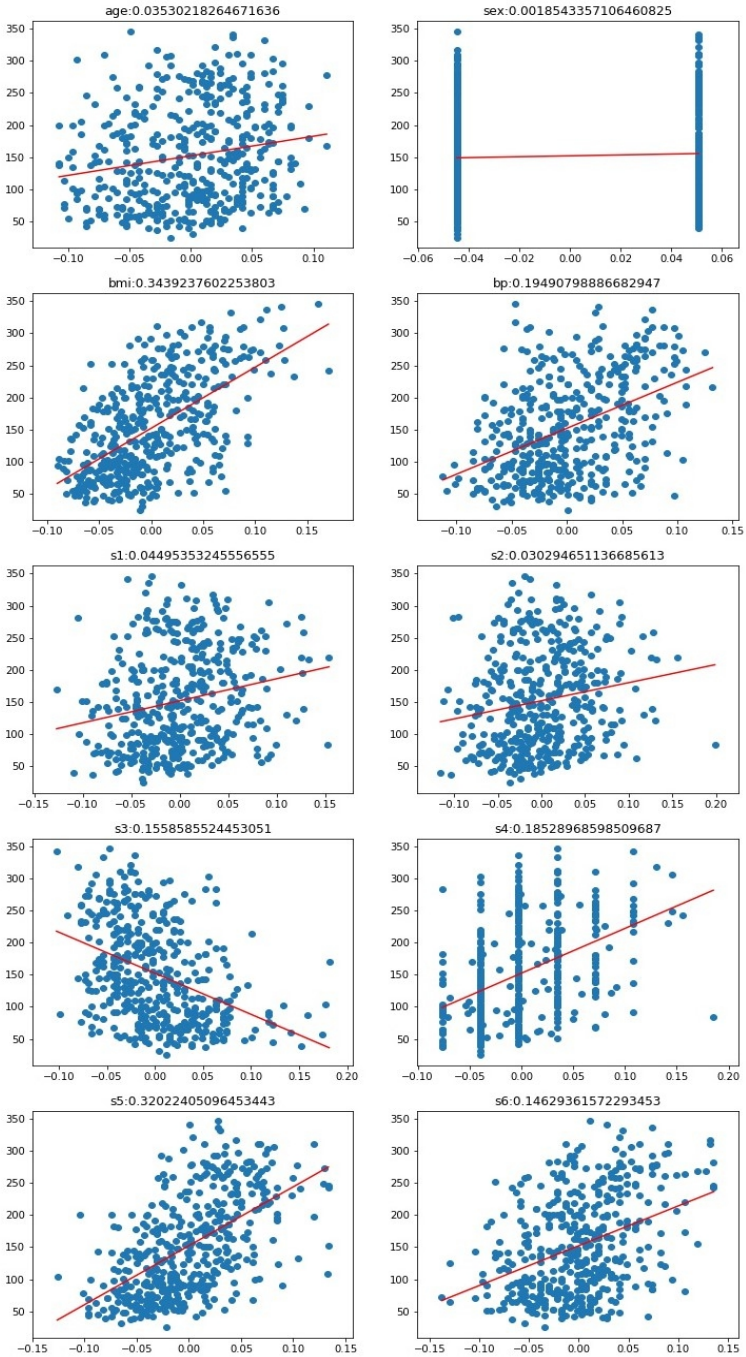

图 1-1 特征与结果的相关性可视化

2、糖尿病预测结果可视化

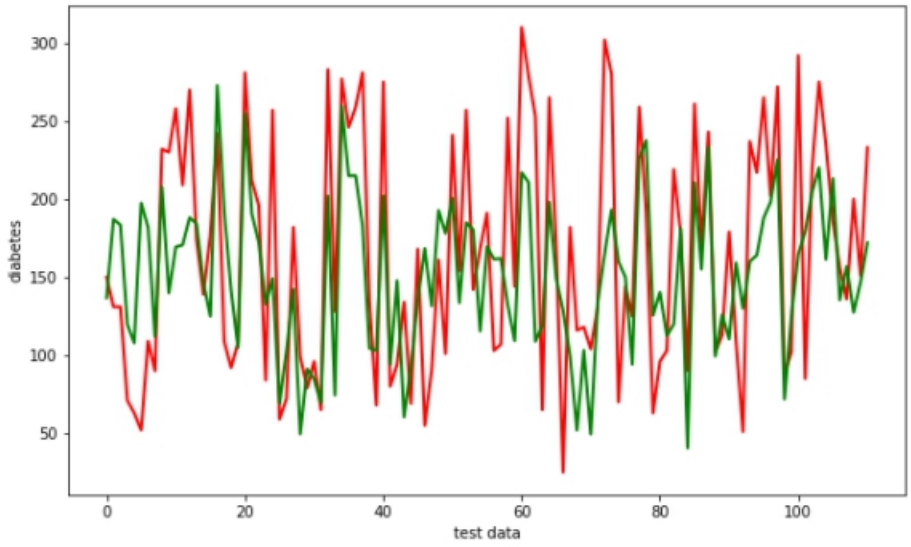

图 1-2 糖尿病预测结果可视化

3、filenum 与 filesize 散点图

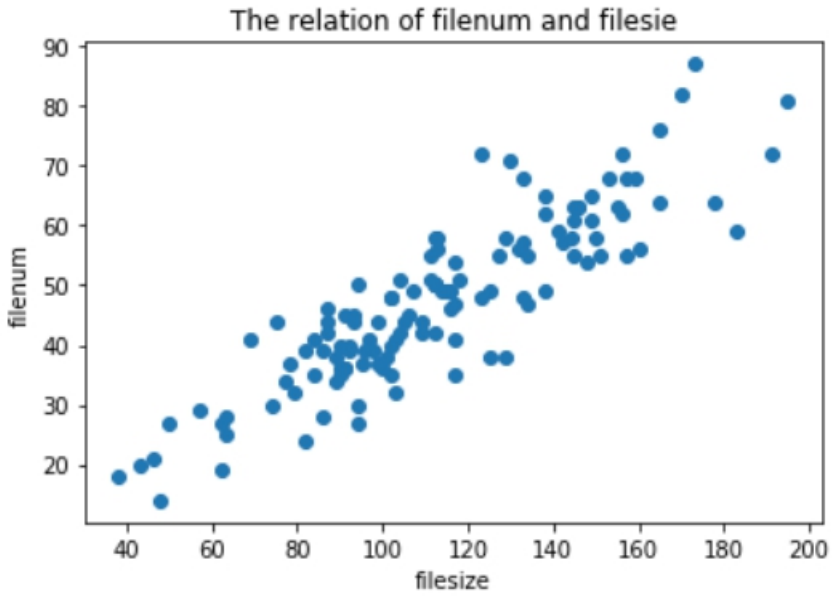

图 1-3 filenum 与 filesize 散点图

4、影厅人数数据集散点图矩阵

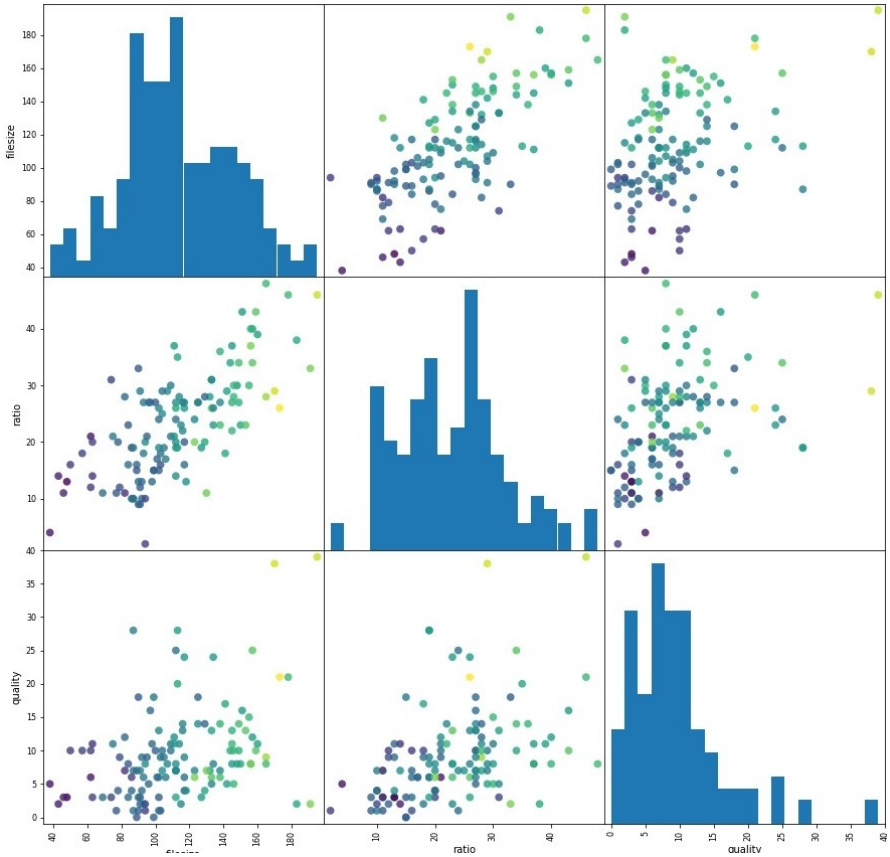

图 1-4 影厅人数数据集散点图矩阵

5、观影人数预测结果可视化

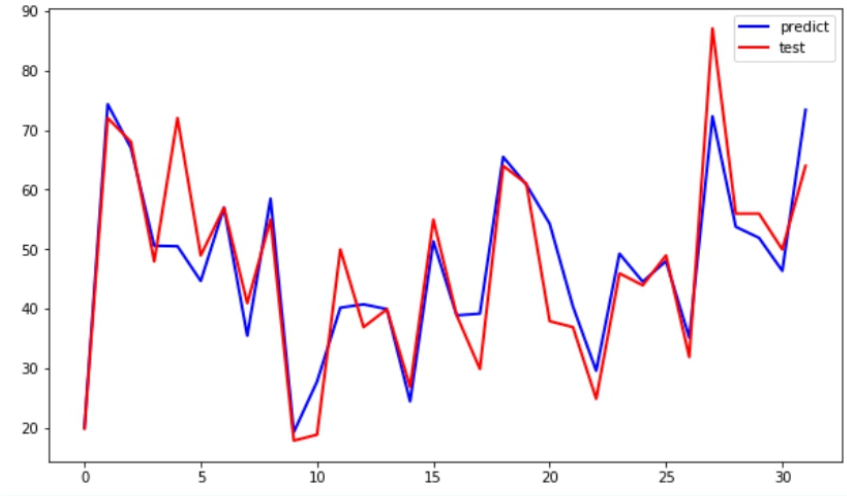

图 1-5 影厅人数预测结果可视化

五、实验总结

1、知识总结

线性模型可以说是机器学习中最基础的、最基本的模型，回归是监督学习的一个重要问题，回归用于预测输入变量和输出变量之间的关系，特别是当输入变量的值发生变化时，输出变量的值也随之发生变化。回归模型正是表示从输入变量到输出变量之间映射的函数优点。在使用线性回归模型时，要考虑过拟合和欠拟合的问题，这点可以在损失函数中加入一个惩罚项解决过拟合问题，使用岭回归或者套索回归。

2、心得体会

通过本次实验了解并掌握了线性模型，对线性模型有了深刻认识，对机器学习问题的一般流程也有了一定的认识，对线性模型的原理很好的理解了，但是自己手动实现线性模型还是有一定的困难，这点希望在以后的学习过程中进一步加强。

实验二 朴素贝叶斯和 SVM

一、实验目的

1.理解并掌握经典的朴素贝叶斯和 SVM 算法。

2.能够基于朴素贝叶斯和 SVM 分类算法分别实现印第安人分类、鸢尾花分类。

3.能够举一反三，基于朴素贝叶斯和 SVM 分类算法实现肿瘤的分类和预测。

二、实验内容

1.对于给定的 2 个例题，基于朴素贝叶斯和 SVM 分别进行印第安人分类、鸢尾花分类等练习。

2.对于给定的 2 个项目，自行编写程序，分别使用朴素贝叶斯和 SVM 分类算法对威斯康星乳腺癌数据集进行肿瘤的分类与预测。

三、实验过程

1、对朴素贝叶斯的认识和理解

朴素贝叶斯是经典的机器学习算法之一，是一种基于概率论的分类算法。朴素贝叶斯原理简单，也很容易实现，多用于文本分类，比如垃圾邮件过滤。朴素贝叶斯方法是在贝叶斯算法的基础上进行了相应的简化，即假定给定目标值时属性之间相互条件独立。也就是说没有哪个属性变量对于决策结果来说占有着较大的比重，也没有哪个属性变量对于决策结果占有着较小的比重。

2、对 SVM 的认识和理解

支持向量机（SVM），通俗来讲，它是一种二类分类模型，其基本模型定义为特征空间上的间隔最大的线性分类器，其学习策略便是间隔最大化，最终可转化为一个凸二次规划问题的求解。SVM 在很多诸如文本分类，图像分类，生物序列分析和生物数据挖掘，手写字符识别等领域有很多的应用，在神经网络出现之前，SVM 可以说在很多领域都是非常好的模型，SVM 可以成功应用的领域远远超出现在已经在开发应用了的领域。

3、对例题的理解

例题 1 是基于朴素贝叶斯实现 Pima 印第安人数据集分类，朴素贝叶斯主要的原理就是先验概率转换为后验概率，通过后验概率完成预测和分类，朴素贝叶斯对于参数的要求较少。有三种朴素贝叶斯算法，先验概率是高斯分布的朴素贝叶斯，主要应用于大部分特征为连续的情况；先验概率为多项式分布的朴素贝叶斯，主要应用于样本特征是多元离散的情况；先验概率为伯努利分布的朴素贝叶斯，主要应用样本特征是二元离散或多元离散。

例题 2 是利用朴素贝叶斯和 SVM 解决鸢尾花分类，训练了一个高斯贝叶斯模型，进行模型训练、预测、评估，之后训练了一个 SVM 模型。可视化结果的时候，使用了主成分分析（PCA），PCA 常用于高维数据的降维，可用于提取数据的主要特征分量。

4、肿瘤分类与预测（朴素贝叶斯）

（1）项目实现思路

对于本次朴素贝叶斯实验，整体思路非常简单，就是机器学习的一般流程，首先载入数据，分析数据，接着配置模型、训练模型、模型预测、模型评估，本次实验的一个小小的难点可能就是最后学习曲线的绘制，其实也不是很难，只需要记录每轮训练过程中的 MSE，最后画出折线图即可。

（2）基本流程

1.导入 sklearn 自带的数据集：威斯康星乳腺肿瘤数据集（load_breast_cancer）。
2.打印数据集键值（keys），查看数据集包含的信息。

3.打印查看数据集中标注好的肿瘤分类（target_names）、肿瘤特征名称（feature_names）。

4.将数据集拆分为训练集和测试集，打印查看训练集和测试集的数据形态（shape）。

5.配置高斯朴素贝叶斯模型。

6.训练模型。

7.评估模型，打印查看模型评分（分别打印训练集和测试集的评分）。

8.模型预测：选取某一样本进行预测。（可以进行多次不同样本的预测）

参考方法：可以打印模型预测的分类和真实的分类，进行对比，看是否一致，如果一致，判断这个样本的肿瘤是一个良性的肿瘤，否则结果相反。 也可以用其他方法进行预测。

9. 扩展（选做）：绘制高斯朴素贝叶斯在威斯康星乳腺肿瘤数据集中的学习曲线。

（3）核心代码

配置朴素贝叶斯模型，这里选择 sklearn 中的高斯朴素贝叶斯模型。并且对模型评估，这里采取十折交叉验证的方法。

```
model=GaussianNB()
model.fit(x_train,y_train.values.ravel())
GaussianNB(priors=None, var_smoothing=1e-09)
score = cross_val_score(model,x_train,y_train.values.ravel(),cv=10,sco
ring='accuracy')
```

画出高斯朴素贝叶斯模型在训练集和测试集上的学习曲线。

```
def plot_learning_curve(algo,X_train,X_test,y_train,y_test):
```

```
    train_score = []
    test_score = []
    for i in range(1,len(X_train)+1):
        algo.fit(X_train[:i],y_train[:i])
        y_train_predict = algo.predict(X_train[:i])
        train_score.append(mean_squared_error(y_train[:i],y_train_pred
ict ))
        y_test_predict = algo.predict(X_test)
        test_score.append(mean_squared_error(y_test,y_test_predict))
    figsize = 11,9
    plt.plot([i for i in range(1,len(X_train)+1)], np.sqrt(train_score
),label = 'Train')
    plt.plot([i for i in range(1,len(X_train)+1)], np.sqrt(test_score)
,label = 'Test')
    plt.axis([0,len(X_train)+1,0,1])
    plt.show()
```

（4）调试过程

本项目的实现较为简单，而且由于朴素贝叶斯算法并没有很多的参数需要设置，所以整个过程的实现很顺利，难点可能是在于学习曲线的绘制，开始的时候对于坐标的设置不合理，导致画出的图像不能很好的反映学习曲线。

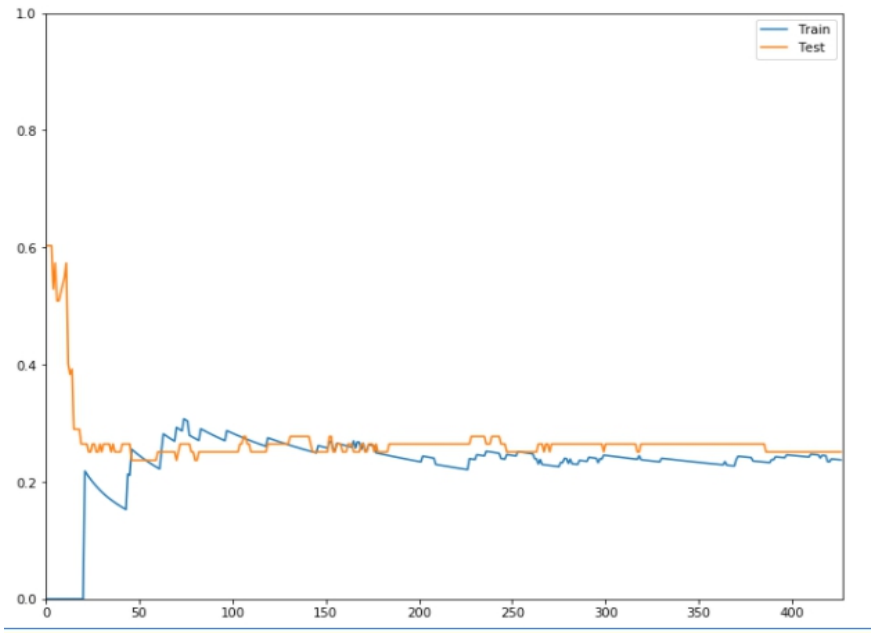

图 2-1 调试过程中绘制的学习曲线

5、肿瘤分类与预测（SVM）

（1）项目实现思路

对于本次项目的实现思路很清楚，由于训练模型、模型预测、模型评估都是 sklearn 提供，所以没有很大的难度。难度在于之前的特征工程，特征工程也是机器学习中一个非常重要的部分，往往特征工程会决定你的模型的最后的效果。此次实验的特征工程主要在于数据的清洗，去掉无用的列（id），将诊断结果字符编码，在特征选取的时候利用热力图呈现字段之间的关系进行特征的选取。

（2）基本流程

1.加载 data 文件夹里的数据集：威斯康星乳腺肿瘤数据集（数据集路径：data/data74924/data.csv）。

2.查看样本特征和特征值，查看样本特征值的描述信息。

3.进行数据清洗（如删除无用列，将诊断结果的字符标识 B、M 替换为数值 0、1 等）。

4.进行特征选取（方便后续的模型训练）。用热力图呈现 features_mean 字段之间的相关性，从而选取特征。

注：（1）热力图中，颜色越浅代表相关性越大。

（2）通过热力图找到相关性大的几个属性,每组相关性大的属性只选一个属性做代表。这样就可以把 10 个属性缩小。

5.进行数据集的划分（训练集和测试集），抽取特征选择的数值作为训练和测试数据。

6.进行数据标准化操作（可采用 Z-Score 规范化数据）。

7.配置模型，创建 SVM 分类器。

8.训练模型。

9.模型预测。

10.模型评估。

（3）核心代码

数据清洗，去掉 id 列，利用 LabelEncoder 对诊断结果编码。

```
features_mean=list(data.columns[2:12])
features_se=list(data.columns[12:22])
features_worst=list(data.columns[22:32])
data=data.drop(['id'],axis=1)
leDiagnosisi=preprocessing.LabelEncoder()
data['diagnosis']=leDiagnosisi.fit_transform(data.diagnosis)
```

绘制热力图，呈现 features_mean 字段之间的相关性。

```
sns.set()
```

```
sns.set_context({ "figure.figsize":( 10,8)})
corr=data[features_mean].corr()
sns.heatmap(corr,annot=True)
plt.show()
```

选取特征，划分数据集。

```
features_select=['radius_mean','texture_mean','smoothness_mean','compa
ctness_mean','symmetry_mean','fractal_dimension_mean']
train_x,test_x,train_y,test_y=train_test_split(data[features_select],d
ata['diagnosis'],test_size=0.25)
```

数据标准化，采用 Z-score 标准化。

```
scaler=StandardScaler()
x_train_scaled=scaler.fit_transform(train_x)
x_test_scaled=scaler.transform(test_x)
```

SVM 模型配置、训练，预测、评估。

```
classifier=SVC(kernel='linear',C=10)
classifier.fit(x_train_scaled,train_y)
prediction=classifier.predict(x_test_scaled)
print('Accuracy:%s' % accuracy_score(test_y, prediction))
```

（4）调试过程

本次实验的调试过程中遇到的一个问题是在规范化数据的时候，有两种办法，分别是 scale()函数和 StandardScaler()函数，二者都可以把数据标准化，处理的过程都是(X-mean)/std。

scale()不能迁移到新的数据集，如果是处理训练集和测试集，只能是把训练集和测试集合起来， 计算出共同的 mean 和 std， 然后 (X-mean) / std，再分成训练集和测试集。这里的 mean 和 std 的计算涉及到了测试集，是训练集和测试集共同的期望和方差

而 StandardScaler()可以迁移到新的数据集，只需要处理训练集，拿训练集的数据计算出均值 x_train_mean, x_train_std 和方差，然后训练集的 X_train 和测试集的 X_test 都执行标准化。

通常情况，都是从整体中采用抽样的方式抽出训练集，这部分训练集可以代替整体，也就是训练集的期望就是整体的期望，测试集标准化的时候，它的期望采用的正是训练集的期望.所以 StandardScaler()才是经常用的方式。

四、实验结果

1、朴素贝叶斯模型的学习曲线

每轮的训练中记录训练数据和测试数据的 MSE，绘制出学习曲线。

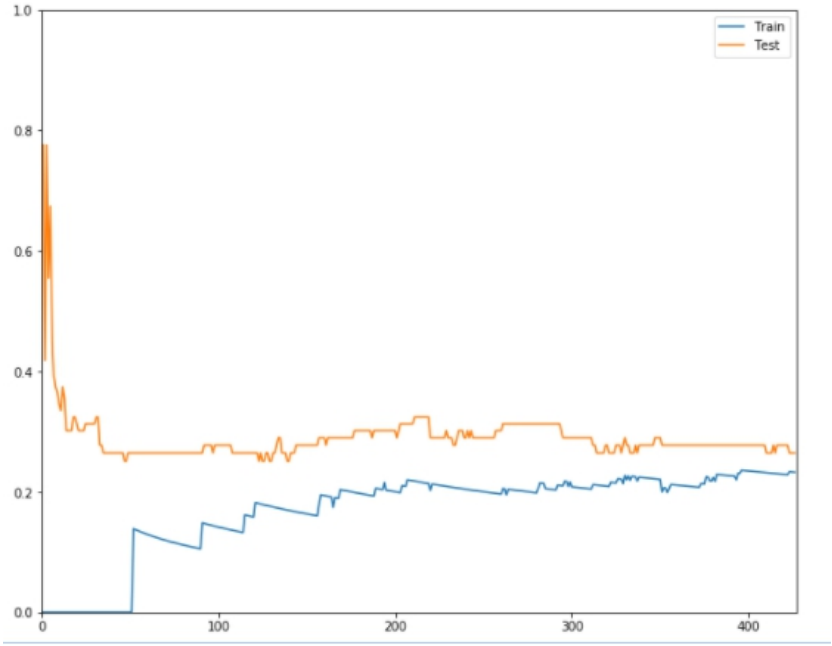

图 2-2 朴素贝叶斯模型学习曲线

2、热力图呈现 features_mean 字段之间的相关性

利用 hetmap 绘制热力图，选取相关性高的特征

图 2-3 热力图

五、实验总结

1、知识总结

朴素贝叶斯算法逻辑简单,易于实现，算法思路很简单，只要使用贝叶斯公式转化即可。分类过程中时空开销小，因为假设特征相互独立，所以只会涉及到二维存储。理论上，朴素贝叶斯模型与其他分类方法相比具有最小的误差率。但是实际上并非总是如此，这是因为朴素贝叶斯模型假设属性之间相互独立，这个假设在实际应用中往往是不成立的，在属性个数比较多或者属性之间相关性较大时，分类效果不好。但特征相关性很小的实际情况还是很多的，所以这个模型仍然能够工作得很好。

SVM 算法和之前学的感知机模型有一定的类似，但是感知机模型只是找到一个可以分类的平面即可，而 SVM 则是要求分类超平面可以使得间隔最大化。显然 SVM 算法明显优于感知机模型。在神经网络流行之前，SVM 几乎就是最好的分类模型，在很多领域都有着很好的应用。

2、心得体会

通过本次实验，了解并掌握了朴素贝叶斯算法和 SVM 模型，可以使用朴素贝叶斯算法和 SVM 解决相关的机器学习问题，但是对于其中的原理推导并不能很好掌握，对于朴素贝叶斯的推导可能较为简单，就是利用朴素贝叶斯公式，将先验概率转换为后验概率。但是对于 SVM 的推导，难度较大，例如对于拉格朗日乘子法和对偶定理的认识还有些模糊，希望在后期的学习过程中，自己可以手动实现朴素贝叶斯算法和 SVM，并不仅仅是调包。

实验三 决策树

一、实验目的

1.理解并掌握经典的决策树分类算法。

2.能够基于决策树分类算法实现鸢尾花分类与预测

3.能够举一反三，基于决策树分类算法实现肿瘤、购买服装等数据的分析与预测。

二、实验内容

1.对于给定的例题，基于决策树分类算法进行鸢尾花分类与预测的练习。

2.对于给定的项目，自行编写程序，使用决策树分类算法对威斯康星乳腺癌数据集、顾客购买服装数据集进行分析与预测。

三、实验过程

1、对决策树的认识和理解

决策树正是做出决策、做出选择，但是选择是根据一定的标准来做出的。决策树的生成算法有 ID3, C4.5 和 CART 等。决策树是一种树形结构，其中每个内部节点表示一个属性上的判断，每个分支代表一个判断结果的输出，最后每个叶节点代表一种分类结果。

决策树是一种十分常用的分类方法，是一种监督学习的算法，监督学习就是给出一组样本，每个样本都有一组属性和一个分类结果，也就是分类结果已知，那么通过学习这些样本得到一个决策树，这个决策树能够对新的数据给出正确的分类。

2、对例题的理解

例题 1 是利用决策树对莺尾花进行分类，莺尾花分类是一个经典的机器学习问题，本次实验利用决策树模型对莺尾花进行分类，决策树模型是一个很好的分类模型，在本次例题中分别采用 ID3 和 CART 算法构建决策树。在本次项目对于决策树的调参有限，只是改了划分属性的评价函数以及最大深度，还有很多参数并没有调整，例如：划分时考虑的特征数、划分属性所需的最小样本、叶子节点的最小样本数等。

例题 2 是根据之前的气象数据来预测将来是否会降雪，此例题采用的是 CART 决策树，并且完全手工实现 CART 决策树，对于一个机器学习的新手，理解决策树的原理相对简单，手工实现决策树还是很困难的，这一点需要在之后的学习过程去深入研究决策树算分，并且多实践，去尝试着自己去实现机器学习相关的算法。

3、肿瘤预测（决策树）

（1）项目实现思路

对于本次实验来说，训练决策树模型并不会很难，难点在于对决策树的调参，本次实验中对决策树的调参，采取网格搜索的方式来寻找最优参数。网格搜索法是指定参数值的一种穷举搜索方法，通过将估计函数的参数通过交叉验证的方法进行优化来得到最优的学习算法。将各个参数可能的取值进行排列组合，列出所有可能的组合结果生成“网格”。然后将各组合用于模型训练，并使用交叉验证对表现进行评估。在拟合函数尝试了所有的参数组合后，返回一个合适的分类器，自动调整至最佳参数组合，可以通过clf.best_params_获得参数值。

（2）基本流程

1.加载 sklearn 自带的威斯康星乳腺癌数据集，探索数据。

2.进行数据集分割。

3.配置决策树模型。

4.训练决策树模型。

5.模型预测。

6.模型评估。

7.参数调优。可以根据评估结果，对模型设置或调整为更优的参数，使评估结果更准确。

（3）核心代码

模型配置，训练模型，模型预测，模型评估

```
clf=tree.DecisionTreeClassifier(criterion='entropy', max_depth=4)
clf.fit(x_train,y_train)
pre=clf.predict(x_test)
print('Accuracy:%s'% accuracy_score(y_test, pre))
```

参数调优，主要采取网格搜索进行调优。

```
clf = tree.DecisionTreeClassifier()
param_grid = {
    'criterion':['gini','entropy'],
    'max_depth':np.arange(3,20)
}
grid = GridSearchCV(clf,param_grid=param_grid,cv = 5)
grid.fit(x_train,y_train)
print(grid.best_params_,
      grid.best_score_,
```

```
      grid.best_estimator_,
      grid.best_index_,
)
```

（4）调试过程

对于本次实验的调试过程主要在于对于参数的调整，而对于决策树来说，相对重要的几个特征就是决策树生成算法以及决策树最大深度，尤其是决策树最大深度，如果决策树最大深度不合适，很容易引起欠拟合和过拟合问题，导致模型效果不佳。在本次实验中采用网格搜索的方法来寻找参数，所以整个调试过程也是十分顺利。

4、顾客购买服装的分析与预测

（1）项目实现思路

本次项目的实现较为简单，就是训练两种决策树模型，在之前的实验中已经都实践过，所以实现起来较为简单，另外本次实验需要可视化决策树，对于这一点，也是采用决策树内部的函数进行实现即可。

（2）基本流程

1.读取顾客购买服装的数据集（数据集路径：data/data76088/3_buy.csv），探索数据。

2.分别用 ID3 算法和 CART 算法进行决策树模型的配置、模型的训练、模型的预测、模型的评估。

3.扩展内容（选做）：对不同算法生成的决策树结构图进行可视化。

（3）核心代码

配置 ID3、CART 两种决策树模型

```
clf=tree.DecisionTreeClassifier(criterion='gini', max_depth=4)
clf.fit(x_train,y_train)
pre=clf.predict(x_test)
print(mean_squared_error(y_test,pre))
print('Accuracy:%s'% accuracy_score(y_test, pre))
```

对决策树进行可视化

```
dot_data = tree.export_graphviz(clf,
                                out_file=None,
                                feature_names=feature_name,
                                class_names ='buy',
                                filled = True,
                                rounded =True )
graph = pydotplus.graph_from_dot_data(dot_data)
```

```
display(Image(graph.create_png()))
```

（4）调试过程

本次实验主要是生成两种决策树，对于这块没有经过太多调试，在决策树的可视化这块查阅了相关资料和博客，进行参考，也较为容易对决策树进行可视化。

四、实验结果

1、决策树可视化

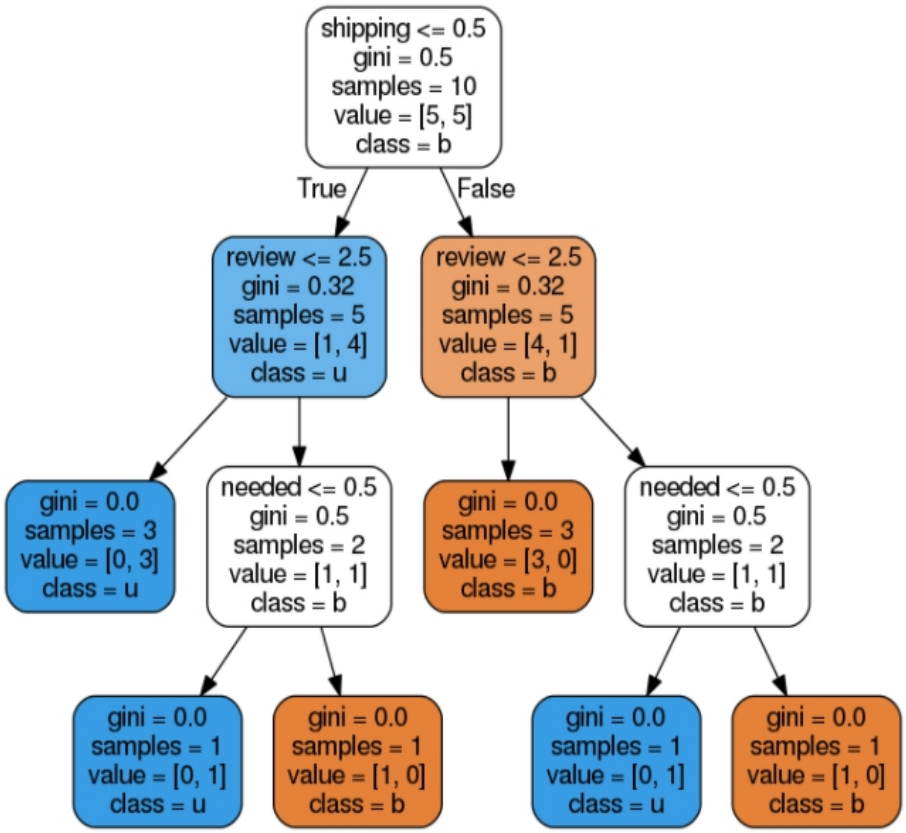

图 3-1 CART 决策树可视化

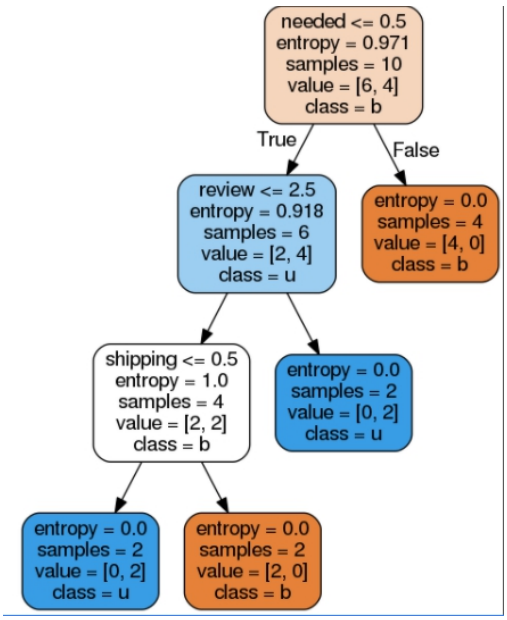

图 3-2 ID3 决策树可视化

五、实验总结

1、知识总结

决策树算法是一种典型的分类方法，首先对数据进行处理，利用归纳算法生成可读的规则和决策树，然后使用决策对新数据进行分析。本质上决策树是通过一系列规则对数据进行分类的过程。决策树的典型算法有 ID3，C4.5，CART 等。

决策树学习的算法通常是一个递归地选择最优特征，并根据该特征对训练数据进行分割，使得对各个子数据集有一个最好的分类的过程。包含特征选择、决策树的生成和决策树的剪枝过程。

决策树算法有很多的优点，决策树算分分类精度高，并且生成的模式很简单，另外决策树算法对噪声数据有很好的健壮性。

2、心得体会

通过本次实验了解并且掌握决策树算法，主要学习了 ID3、C4.5、CART 三种决策树生成算法，并且对三种决策树算法的原理有了一定的了解，可以借助 sklearn、来构建这三种决策树生成算法的模型。对于决策树算法的构建，并不能自己独立手动构建，这是在之后的学习过程中需要加强的地方，需要在之后的学习过程中尝试手动实现机器学习算法，加强自己的动手能力。

实验四 聚类算法

一、实验目的

1.理解并掌握聚类算法模型。

2.能够基于 K-Means 聚类算法模型实现莺尾花分类、基于经纬度的城市聚类。

3.能够举一反三，基于 K-Means 聚类算法实现果汁饮料聚类分析。

二、实验内容

1.对于给定的例题，基于 K-Means 聚类算法进行莺尾花分类、基于经纬度的城市聚类等练习。

2.对于给定的项目，自行编写程序，使用 K-Means 聚类算法实现果汁饮料聚类分析。

三、实验过程

1、对聚类算法及 K-means 的认识和理解

聚类是一种机器学习技术，它涉及到数据点的分组。给定一组数据点，可以使用聚类算法将每个数据点划分为一个特定的组。理论上，同一组中的数据点应该具有相似的属性和/或特征，而不同组中的数据点应该具有高度不同的属性和/或特征。聚类是一种无监督学习的方法，是许多领域中常用的统计数据分析技术。

K-means 是最普及的聚类算法，算法接受一个未标记的数据集，然后将数据聚类成不同的组，K-means 是一个迭代算法，通过不断迭代将数据分组。

2、对例题的理解

例题 1 是针对莺尾花数据的分类，莺尾花数据是有标签的，但是 K-means 是无监督学习，是不需要标签的，这里只需要假设莺尾花数据集没有标签即可，K-Means 通过平均的方法可以以丛聚的方式将数据分类，但是在训练时并没有放入标签答案，所以程序在预测之后，需要调整。

例题 2 是典型的无监督学习问题，对于给定城市的经纬度将数据进行分类，利用 K-means 可以很好、很方便解决这个问题，这样可以对所给出的城市分类，并且进行可视化显示。

3、不同果汁饮料的聚类

（1）项目实现思路

对于本次项目的实现，首先先是加载果汁数据集，观察数据的特点，完后需要把原始数据以散点图显示方便之后 k 值的选择，对待选的几个 k 值分别建立聚类模型训练，评估模型选出最优的，并且进行可视化显示结果，包括对于聚类中心的显示。

（2）基本流程

1.加载数据集，读取数据，探索数据。

2.样本数据转化（可将 DataFrame 格式的数据转化为数组形式），并进行可视化（绘制散点图），观察数据的分布情况，从而可以得出 k 的几种可能取值。

3.针对每一种 k 的取值，进行如下操作：

（1）进行 K-Means 算法模型的配置、训练。

（2）输出相关聚类结果，并评估聚类效果。 这里可采用 CH 指标来对聚类有效性进行评估。在最后用每个 k 取值时评估的 CH 值进行对比，可得出 k 取什么值时，聚类效果更优。

注：这里缺乏外部类别信息，故采用内部准则评价指标（CH）来评估。

（3）输出各类簇标签值、各类簇中心，从而判断每类的果汁含量与糖分含量情况。

（4）聚类结果及其各类簇中心点的可视化（散点图），从而观察各类簇分布情况。（不同的类表明不同果汁饮料的果汁、糖分含量的偏差情况。）

4.设置 k 一定的取值范围，进行聚类并评价不同的聚类结果，。设置 k 的取值范围；对不同取值 k 进行训练；计算各对象离各类簇中心的欧氏距离，生成距离表；提取每个对象到其类簇中心的距离，并相加；依次存入距离结果；绘制不同 k 值对应的总距离值折线图。

（3）核心代码

将数据转换为 array 类型，并且利用 pyplot 画出散点图，观察数据分布。

```
data=np.array(data)
plt.figure(figsize=(9,6))
plt.xlabel('juice')
plt.ylabel('sweet')
plt.scatter(data[:,0],data[:,1], s=50, c='r', marker='o')
plt.show()
```

根据数据分布，选取了 k 的范围为（2,6），训练模型，记录比较每个模型的 CH 值，对于这一步，采用画出不同 k 值训练的模型的 CH 得分的折线图来显示实验结果。

```
score_all=[]
li=range(2,6)
for i in range(2,6):
    y_pred = KMeans(n_clusters=i, random_state=9).fit_predict(data)
    plt.scatter(data[:, 0], data[:, 1], c=y_pred)
    plt.show()
    score=metrics.calinski_harabaz_score(data, y_pred)
    score_all.append(score)
```

```
    print(score)
plt.plot(li,score_all)
plt.show()
```

根据上面的实验结果可以发现当 k 取 4 时模型效果最好，所以选择 k=4 进行模型训练预测，并且可视化聚类结果及其各类簇中心点，输出预测值以及聚类中心。

```
markers = ['*', 's', 'x', 'o']      # 标记样式列表
colors = ['r', 'g', 'm',  'b']  # 标记颜色列表
labels = estimator.labels_
plt.figure(figsize=(9, 6))
plt.xlabel('juice', fontsize=18)
plt.ylabel('sweet', fontsize=18)
for i in range(4):
    members = labels == i
    plt.scatter(
        data[members, 0],
        data[members, 1],
        s=200,
        marker = markers[i],
        c = colors[i]
    )
centers = estimator.cluster_centers_
plt.scatter(centers[:,0], centers[:, 1], c='black', s=200, alpha=0.7);
plt.show()
print(estimator.labels_)
print(estimator.cluster_centers_)
```

（4）调试过程

对于本次项目的调试主要是针对不同 k 值进行调参，对于这个调参过程，我采用可视化的办法进行调参，并且对于不同 k 值的聚类模型，对其评价，这里采用 CH 指标进行评价。

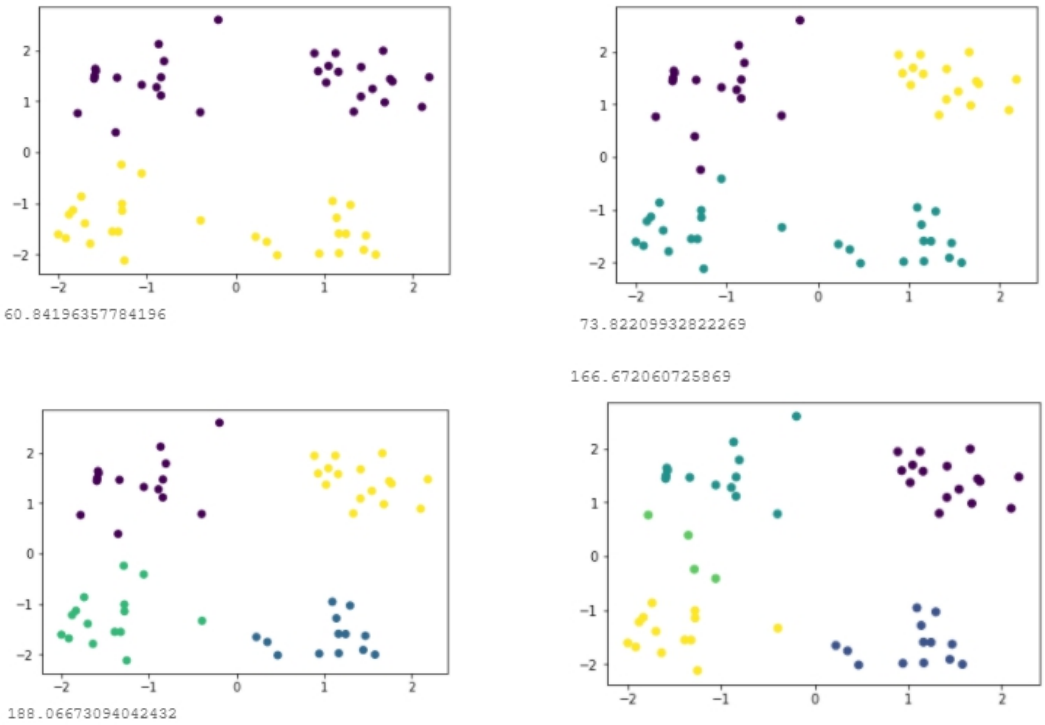

图 4-1 对 k 值的调参

四、实验结果

1、原始数据分布

利用散点图对原始数据分析，根据散点图选出可能的 k。

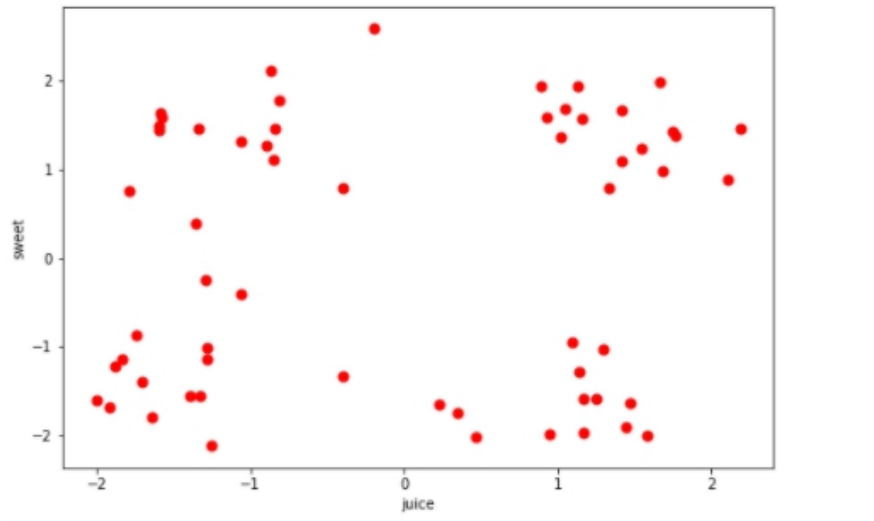

图 4-2 原始数据分布图

2、k 值-CH 值折线图

根据不同的 k 值对应不同的 CH 得分，画出折线图。

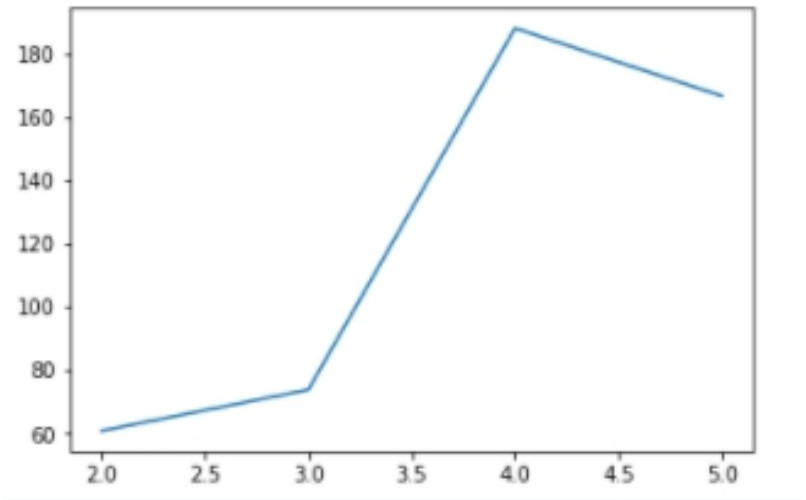

图 4-3 原始数据分布图

3、选取 k=4，可视化分类结果

由上述实验可知，当 k=4 时，模型的预测效果最好，所以选取 k=4，可视化分类结果，并且标出每一类的中心。

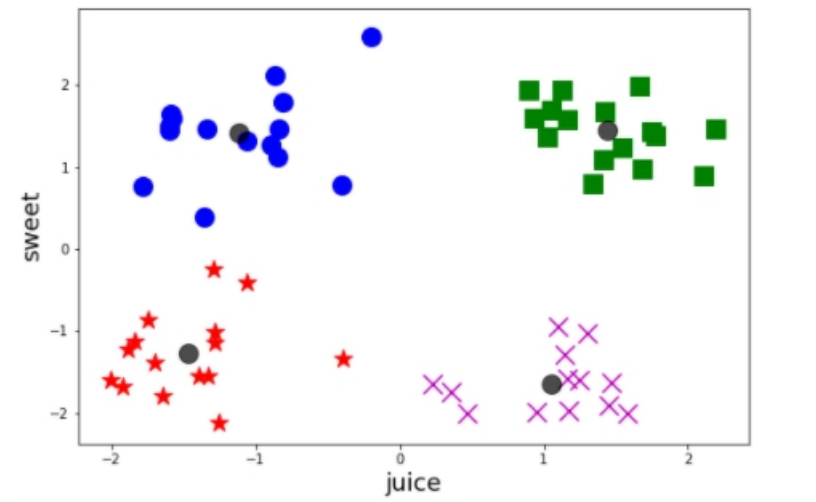

图 4-4 k=4 的分类结果可视化

五、实验总结

1、知识总结

k-means 是一种迭代求解的聚类分析算法，预将数据分为 K 组，则随机选取 K 个对象作为初始的聚类中心，然后计算每个对象与各个种子聚类中心之间的距离，把每个对象分配给距离它最近的聚类中心。聚类中心以及分配给它们的对象就代表一个聚类。每分配一个样本，聚类的聚类中心会根据聚类中现有的对象被重新计算。这个过程将不断重复直到满足某个终止条件。终止条件可以是没有（或最小数目）对象被重新分配给不同的聚类，没有（或最小数目）聚类中心再发生变化，误差平方和局部最小。

k-means 在大数据的条件下，会耗费大量的时间和内存。所以在对模型调参的时候可以减少聚类的数目 K，减少样本的特征维度。比如说，通过 PCA 等进行降维。考察其他的聚类算法，去测试不同聚类算法的性能。

2、心得体会

通过本次实验了解并且掌握了聚类学习算法中的 K-means 算法，这是学习机器学习以来第一次接触非监督学习，对于非监督学习算法有了一个初步的了解，在之后的学习过程中希望进一步深入研究相关的非监督学习算法，例如高斯混合模型（GMM）等算法模型。

实验五 AdaBoost

一、实验目的

1.理解并掌握集成学习中 AdaBoost 算法。

2.能够基于 AdaBoost 算法实现莺尾花分类。

3.能够举一反三，基于 AdaBoost 算法实现肿瘤预测。

二、实验内容

1.对于给定的例题，基于 AdaBoost 集成学习算法进行莺尾花分类的练习。

2.对于给定的项目，自行编写程序，使用 AdaBoost 集成学习算法实现肿瘤预测。

三、实验过程

1、对集成学习算法的认识和理解

集成学习就是用多重或多个弱分类器结合为一个强分类器，从而达到提升分类方法效果。集成学习并不算是一种分类器，而是一种分类器结合的方法。

Bagging 和 Boosting 是两种著名的集成学习方法，Bagging 是从原来的数据集中放回取出的数据，同时保持数据集的规模不变。用新的数据集训练弱分类器。重复上述过程多次，取平均值或者采用投票机制。而 Boosting 是首先训练一个弱分类器，这个分类器基于错误样本训练，给予这个分类器一个权重，重复上述过程，直到分类器 性能达到某一指标；最后，把这些分类器乘上相应的权重全部加起来，就得到了最后的强分类器。

AdaBoost 它的自适应在于：前一个基本分类器分错的样本会得到加强，加权后的全体样本再次被用来训练下一个基本分类器。同时，在每一轮中加入一个新的弱分类器，直到达到某个预定的足够小的错误率或达到预先指定的最大迭代次数 。

2、对例题的理解

本例题是用 AdaBoost 算法来进行对鸢尾花分类，针对鸢尾花数据，之前已经通过很多种方法对其进行分类预测。在本次实验例题中，首先是构造了一个 AdaBoost 模型，但是并没有进行调参，参数全部采用默认值，最后模型的准确率只有百分之七十左右。在进行简单地调参后，例如调整基分类器的数量（基分类器提升次数）或者学习率，模型的准确率可以达到百分之九十六左右。

在进行模型调参的时候主要调参的是基分类器的数量、学习率、基分类器等。基分类器的数量默认为 50，这个值过大，模型容易引起过拟合；值过小，模型容易欠拟合。学习率表示梯度收敛速度，当分类器迭代次数较少时，学习率可以小一些，当迭代次数较多时，学习率可以适当放大。在调参时可以利用网格搜索进行调参，设置一个参数范围，利用网格搜索找出最好的参数。

相对于之前用 SVM、决策树等分类模型对莺尾花分类，我认为利用 Adaboost 对莺尾花进行分类的效果最好，可以看出集成学习算法的优点，利用多个弱分类器集成出一个强分类器，强分类器有着更好的预测准确度。

3、肿瘤预测（AdaBoost）

（1）项目实现思路

对于本次项目，主要是考察对于集成学习方法 Adaboost 的掌握，所以在本次项目实现的时候，首先先是对数据集进行一个加载，这个乳腺癌数据集在之前的实验中已经用了很多次，所以在此并不再对此数据集进行一个深入的分析。

此项目的实现主要利用 AdaBoost 算法进行一个模型的构建和训练、评估。整个流程和一般的机器学习方法一致。重点在于对 AdaBoost 的调参过程，因为模型都是直接调用的，而参数是需要你自己去调整的，调参的好坏直接决定了模型的预测准确率。

（2）基本流程

1.加载 sklearn 自带的数据集，使用 DataFrame 形式探索数据。

2.划分训练集和测试集，检查训练集和测试集的平均癌症发生率。

3.配置模型，训练模型，模型预测，模型评估。

（1）构建一棵最大深度为 2 的决策树弱学习器，训练、预测、评估。

（2）再构建一个包含 50 棵树的 AdaBoost 集成分类器（步长为 3），训练、预测、评估。

参考：将决策树的数量从 1 增加到 50，步长为 3。输出集成后的准确度。

（3）将（2）的性能与弱学习者进行比较。

4.绘制准确度的折线图，x 轴为决策树的数量，y 轴为准确度。

（3）核心代码

利用 sklearn 中的 DecisionTreeClassifier 构建一棵最大深度为 2 的决策树弱学习器，训练、预测、评估。

```
clf=tree.DecisionTreeClassifier(criterion='gini', max_depth=2)
clf.fit(x_train,y_train)
pre=clf.predict(x_test)
print('Accuracy:%s'% accuracy_score(y_test, pre))
```

用 for 循环控制决策树的数量，从 1 增加到 50，步长为 3，记录每次的决策树数量以及对应的模型准确率，为下面画折线图准备。

```
score_all=[]
tree_num=[]
for i in range(1,50,3):
    model = AdaBoostClassifier(n_estimators=i)
```

```
    model.fit(x_train,y_train )
    predictions = model.predict(x_test)
    tree_num.append(i)
    score=accuracy_score(y_test, predictions)
    score_all.append(score)
    print('基分类器数量：',i,' 准确率：',score)
```

绘制准确度的折线图，x 轴为决策树的数量，y 轴为准确度。

```
plt.plot(tree_num, score_all)
plt.show()
```

（4）调试过程

对于本次实验，并没有特别大的难度，在实验过程中相对比较难的可能就是将决策树的数量从 1 增加到 50，步长为 3。输出集成后的准确度。这点也没有很大的难度，只需 for 循环控制基分类器的数量即可。

对最后的准确率和决策树数量折线图分析，可以看出准确率在一定范围内是随着决策树数量的增加不断增加的，但是最后趋于一个值。从中可以得出，在集成模型中基分类器的数量不能太多也不能太少，太少会发生欠拟合，太多会发生过拟合，模型的泛化能力不好。

四、实验结果

1、AdaBoost 训练数据预测准确度以及决策树数-准确度折线图

将决策树的数量从 1 增加到 50，步长为 3。输出集成后的准确度。绘制准确率-决策树数量折线图。

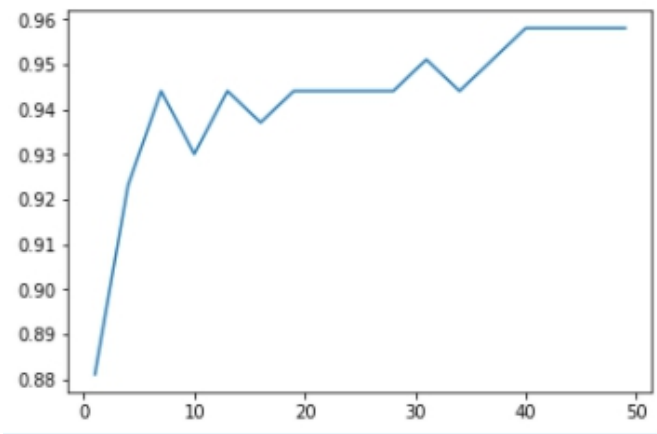

图 5-1 准确率-决策树数量折线图

五、实验总结

1、知识总结

Boosting 方法训练基分类器时采用串行的方式，各个基分类器之间有依赖。它的基本思路是将基分类器层层叠加，每一层在训练的时候，对前一层基分类器分错的样本，给予更高的权重。测试时，根据各层分类器的结果的加权得到最终结果。

Adaboost 算法基本原理就是将多个弱分类器（弱分类器一般选用单层决策树）进行合理的结合，使其成为一个强分类器。

Adaboost 采用迭代的思想，每次迭代只训练一个弱分类器，训练好的弱分类器将参与下一次迭代的使用。也就是说，在第 N 次迭代中，一共就有 N 个弱分类器，其中 N-1 个是以前训练好的，其各种参数都不再改变，本次训练第 N 个分类器。其中弱分类器的关系是第 N 个弱分类器更可能分对前 N-1 个弱分类器没分对的数据，最终分类输出要看这 N 个分类器的综合效果。

Bagging 与 Boosting 的串行训练方式不同，Bagging 方法在训练过程中，各基分类器之间无强依赖，可以进行并行训练。思想就是从总体样本当中随机取一部分样本进行训练，通过多次这样的结果，进行投票获取平均值作为结果输出，这就极大可能的避免了不好的样本数据，从而提高准确度。因为有些是不好的样本，相当于噪声，模型学入噪声后会使准确度不高。

2、心得体会

通过本次实验很好地了解到了集成学习算法，之前并没有接触过集成学习算法，通过这次实验对集成学习算法了有了一个很好的认识。但是在学习集成学习算法的时候，只是对原理有了大概的认识，以及学会了用 sklearn 中封装好的集成学习算法模型来训练模型、预测数据，对其中深入的原理并没有很好掌握，希望在之后可以自己动手去手动实现一下 AdaBoost 算法,这样才可以加深对集成学习算法的理解。另外由于课时有限，课上并没有对另一种集成学习算法 Bagging 进行讲解,只是对其进行了一个简单的介绍，希望在之后的学习过程中可以对 Bagging 也进行一个系统化的学习。

实验六 神经网络

一、实验目的

1. 理解并掌握神经网络模型。

2. 能够基于神经网络模型实现手写数字识别。

3. 能够举一反三，基于神经网络模型实现肿瘤预测与分析。

二、实验内容

1. 对于给定的例题，基于 BP 神经网络算法进行手写数字识别的练习。

2. 对于给定的项目，自行编写程序，使用 BP 神经网络实现肿瘤预测与分析。

3. 扩展内容：学习扩展实验 4、5、6 中的深度神经网络（DNN）、卷积神经网络（CNN）、循环神经网络（RNN）内容，对手写数字识别、猫狗分类、机器阅读理解项目进行练习。

三、实验过程

1、对神经网络的认识和理解

神经网络是一种模拟人脑的神经网络以期能够实现类人工智能的机器学习技术。人脑中的神经网络是一个非常复杂的组织。成人的大脑中估计有 1000 亿个神经元之多。

神经网络类似与人大脑中的神经网络，一个神经网络的逻辑单元类似与生物的一个神经元。神经网络的一般结构是由输入层、隐藏层（神经元）、输出层构成的，隐藏层可以是 1 层或者多层叠加，层与层之间是相互连接的。

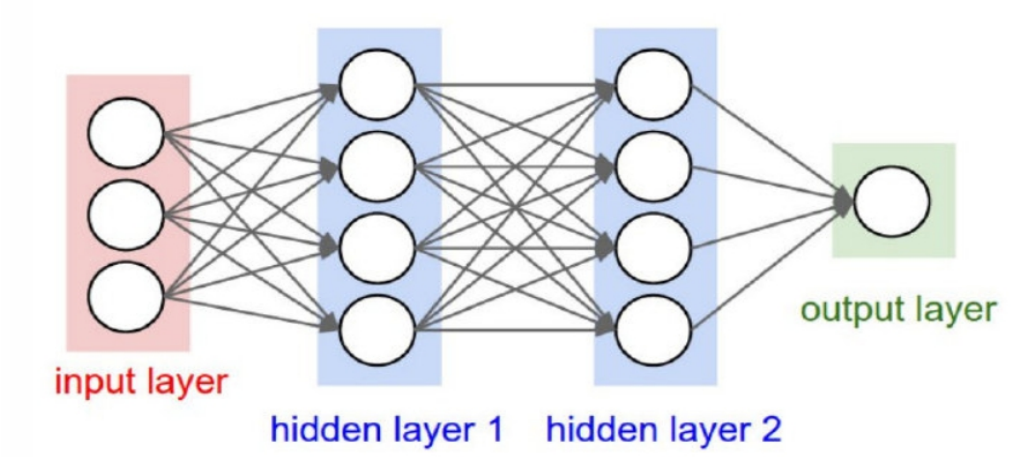

图 6-1 神经网络示意图

2、对例题的理解

例题是针对经典的 mnist 手写数字数据集进行数字图片的预测，该数据集主要包含两个部分，一个是数据的分类标签，另一个是数据本身。数据集总共包括 60000 行的训练数据集（mnist.train）和 10000 行的测试数据集（mnist.test）。每一张图片包含 28X28 个像素点。可以用一个数字数组来表示这张图片，从这个角度来看，mnist 数据集的图片就是在 784 维向量空间里面的点。

在这个例题中对于神经网络的建立是通过 sklearn 中的包来建立的，建立的神经网络有两个 100 个节点的隐藏层，是一个非常简单的神经网络，预测精度可以达到百分之九十二左右，已经是一个相当高的分数。但是还是有很大的提升空间，我在使用 keras 中的 CNN 对 mnist 手写数字数据集进行了分类，预测精度可以达到百分之九十八左右，可见,对于图片的分类,使用卷积神经网络比普通的 BP 神经网络有着更好地预测准确率。

```
model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 26, 26, 32)        320
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 11, 11, 64)        18496
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 3, 3, 64)          36928
_________________________________________________________________
flatten_1 (Flatten)          (None, 576)               0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                36928
_________________________________________________________________
dense_2 (Dense)              (None, 10)                650
=================================================================
Total params: 93,322
Trainable params: 93,322
Non-trainable params: 0
_________________________________________________________________
```

图 6-2 构建的用于 mnist 的卷积神经网络

3、肿瘤预测与分析（神经网络）

（1）项目实现思路

对于本次项目的实现，首先是对数据集进行分析，本次项目的数据集是 sklearn 中的乳腺癌数据集，在之前的实验中已经进行过很好的分析，所以本次项目中并没有进行深度分析。

本次项目重点是建立神经网络模型，在本次项目中采用的是 sklearn 中自带的神经网络模型，只需要自己调参即可。在模型评估的时候需要计算出每一类的预测准确率、召回率和 F1 得分,这些都是根据混淆矩阵计算出的,对于这些评估指标应该很好的掌握。

本项目的一个难点是画 3D 图，由于是 3D 图，那么自热而然想到要选取 3 个特征，但是不能随便选取，于是采取了 PCA（主成分分析）先对数据进行了一个降维，最后再画出 3D 图。

（2）基本流程

1.加载 sklearn 自带的数据集，探索数据。

2.划分训练集与测试集。

3.建立 BP 模型（评估后可进行调参，从而选择最优参数）。

4.进行模型训练。

5.进行模型预测，对真实数据和预测数据进行可视化（用 Axes3D 绘制 3d 散点图）。

6.进行模型评估，并进行预测结果指标统计（统计每一类别的预测准确率、召回率、F1 分数）。

7.计算混淆矩阵，并用热力图显示。

（3）核心代码

建立 BP 模型，进行模型训练，模型预测以及对神经网络模型进行调参。

```
clf= MLPClassifier(random_state=1, max_iter=300).fit(x_train, y_train)
y_pre=clf.predict(x_test)
clf.score(x_test, y_test)
```

进行模型预测，对真实数据和预测数据进行可视化（用 Axes3D 绘制 3d 散点图）。首先先利用 sklearn 中的主成分分析 PCA 对数据进行降维,把三十个特征降维三个特征，完后才可以画 3D 图，注意在颜色 c 属性的值应该为一维数据，但是 label 数据是二维的要先进行一个转换才可以，在项目中是利用 flatten（）进行转换。

```
pca=PCA(n_components=3)
data=pca.fit_transform(x_test)
target=np.array(y_test).flatten()
fig = plt.figure(figsize=(15,6))
```

```
ax = Axes3D(fig)
x = data[:,0]
y = data[:,1]
z = data[:,2]
ax.view_init(elev=45, azim=55)
ax.scatter(x,y,z,c=target,s=100)
plt.show()
```

进行模型评估，并进行预测结果指标统计（统计每一类别的预测准确率、召回率、F1 分数）。对于计算每一类的预测准确率、召回率、F1 分数都是根据混淆矩阵来计算的。

某个类 i 的精度：元素 confmatrix（i，i）除以下标为 i 的纵坐标元素和，公式为：

$$label\_i\_prediction = \frac{confmartix(i,i)}{\sum_j confmartix(j,i)}$$

某个类 i 的召回率：元素 confmatrix（i，i）除以下标为 i 的横坐标元素和,公式为：

$$label\_i\_recall = \frac{confmartix(i,i)}{\sum_j confmartix(i,j)}$$

F1 值：总体样本（或某个类）的精度和召回率满足如下：

$$F1 = \frac{2 \times prediction \times recall}{prediction + recall}$$

```
def calculate_label_prediction(confMatrix,labelidx):
    #计算某一个类标预测精度：该类被预测正确的数除以该类的总数
    label_total_sum = confMatrix.sum(axis=0)[labelidx]
    label_correct_sum = confMatrix[labelidx][labelidx]
    prediction = 0
    if label_total_sum != 0:
        prediction = round(100*float(label_correct_sum)/float(label_to
tal_sum),2)
    return prediction
```

```
def calculate_label_recall(confMatrix,labelidx):
    #计算某一个类标的召回率：
    label_total_sum = confMatrix.sum(axis=1)[labelidx]
    label_correct_sum = confMatrix[labelidx][labelidx]
    recall = 0
    if label_total_sum != 0:
        recall = round(100*float(label_correct_sum)/float(label_total_
sum),2)
    return recall

def calculate_f1(prediction,recall):
    if (prediction+recall)==0:
        return 0
    return round(2*prediction*recall/(prediction+recall),2)
```

计算混淆矩阵，并用热力图显示。混淆矩阵的建立直接可以利用 sklearn 中的 confusion_matrix 来生成，并使用 seaborn 中的 hetmap 生成热力图。

```
from sklearn.metrics import confusion_matrix
print('混淆矩阵：')
matr=confusion_matrix(y_true=y_test,y_pred=y_pre)
import seaborn as sns
sns.set()
sns.set_context({ "figure.figsize":( 3,3)})
sns.heatmap(matr,annot=True)
plt.show()
```

（4）调试过程

本次项目对于神经网络的建立并没有很大的难度，其中整个项目的一个难点是用 Axes3D 绘制 3d 散点图。首先是对于 x、y、z 轴取值的问题，数据集中总共用三十个特征，但是 3d 散点图只能用 3 个特征，所以采用主成分分析（PCA），PCA 就是当特征很多的时候，数据维数压缩，尽可能降低原始数据的维数。所以利用 PCA 降维到 3 维。

另外一点在调试的时候遇到的问题就是数据维度不匹配的问题，由于 label 是二维的，在绘图的时候，不同的 label 需要显示出不同的颜色，所以直接让颜色属性 c=label，但是 label 是二维的，需要转换为一位数据。这点转换数据维度的方法很多，我在这里采用了 flatten()方法直接把数据转换为一维。

```
   2364         zs = _backports.broadcast_to(zs, len(xs))
/opt/conda/envs/python35-paddle120-env/lib/python3.7/site-packages/matplotlib/__init__.py in inner(ax, *a
**kwargs)
   1865                         "the Matplotlib list!)" % (label_namer, func.__name__),
   1866                         RuntimeWarning, stacklevel=2)
-> 1867             return func(ax, *args, **kwargs)
   1868
   1869         inner.__doc__ = _add_data_doc(inner.__doc__,
/opt/conda/envs/python35-paddle120-env/lib/python3.7/site-packages/matplotlib/axes/_axes.py in scatter(se
x, y, s, c, marker, cmap, norm, vmin, vmax, alpha, linewidths, verts, edgecolors, **kwargs)
   4291                 raise ValueError("c of shape {} not acceptable as a color "
   4292                                  "sequence for x with size {}, y with size {}"
-> 4293                                  .format(c.shape, x.size, y.size))
   4294         else:
   4295             colors = None  # use cmap, norm after collection is created
ValueError: c of shape (143, 1) not acceptable as a color sequence for x with size 143, y with size 143
```

图 6-3 数据大小不匹配错误

四、实验结果

1、3D 散点图可视化真实数据和预测数据

先对数据进行降维，完后画出真实数据和预测数据的 3D 可视化图。

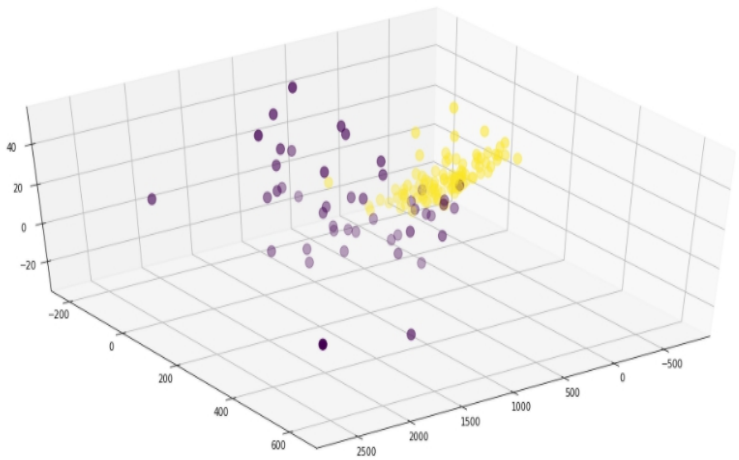

图 6-4 真实数据可视化 3D 图

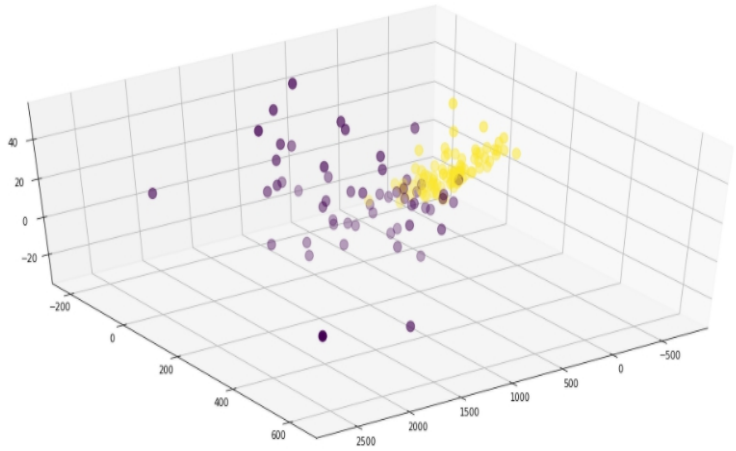

图 6-5 预测数据可视化 3D 图

2、混淆矩阵的热力图

求出混淆矩阵，并且画出热力图，图中颜色越浅，代表的数值越大。

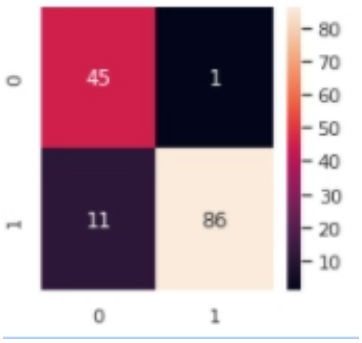

图 6-6 混淆矩阵热力图

五、实验总结

1、知识总结

神经网络是一种从信息处理角度对人脑神经元网络进行抽象从而建立的某种简单模型，按不同的连接方式组成不同的网络。在神经网络中，神经元是计算的基本单元，也被称为节点或单元。它接受其他节点或外部的输入，在计算后产生输出。每两个节点间的连接都代表一个对于通过该连接信号的加权值，称之为权重(w)。该节点将一个函数 f（定义如下）作用于输入的加权和。

混淆矩阵（confusion matrix）衡量的是一个分类器分类的准确程度。混淆矩阵的每一列代表了预测类别 ，每一列的总数表示预测为该类别的数据的数目；每一行代表了

数据的真实归属类别 ，每一行的数据总数表示该类别的数据实例的数目。

对于评估模型好坏的指标，准确率是分类正确的样本占总样本个数的比例，精确率指模型预测为正的样本中实际也为正的样本占被预测为正的样本的比例。召回率指实际为正的样本中被预测为正的样本所占实际为正的样本的比例。F1 score 是精确率和召回率的调和平均值。

2、心得体会

通过本次实验了解并掌握了如何编写简单的神经网络模型，对于神经网络模型还是只有一个简单的理解，并没有深入理解其中的原理，在之后的学习过程中要再去深入研究，并且尝试自己动手手写一个神经网络，包括其中的损失函数、目标函数、梯度下降等。

本次实验使用的神经网络是 sklearn 中封装好的，但是一般深度学习领域并不会使用 sklearn 中的神经网络，更多的使用框架来搭建神经网络，例如：Tensorflow、Pytorch 等，在今后的学习过程中希望可以去学习一种深度学习框架，并且掌握一种深度学习框架。

深度学习在目前有着很多的应用领域，例如：语音识别、计算机视觉、自然语言处理等，因为研究生阶段的方向很多都是这方面的研究，所以我希望在今后的学习过程中寻找到自己感兴趣的研究方向，并学习研究相关的知识。

这次是最后一次实验课，机器学习这门课由于课时有限，课堂老师传授的知识也有限，希望通过这门课，可以对人工智能有了一个很好的入门，有利于之后对人工智能领域更深的研究。

封面设计： 贾丽

地 址：中国河北省秦皇岛市河北大街 438 号
邮 编：066004
电 话：0335-8057068
传 真：0335-8057068
网 址：http://jwc.ysu.edu.cn