《机器学习》期中作业报告

1854084 徐晨龙 交通运输工程学院

在本次作业中大部分的代码都是来自于网上，其中本人主要所做的部分是在已有代码的工具上进行流程的整合。

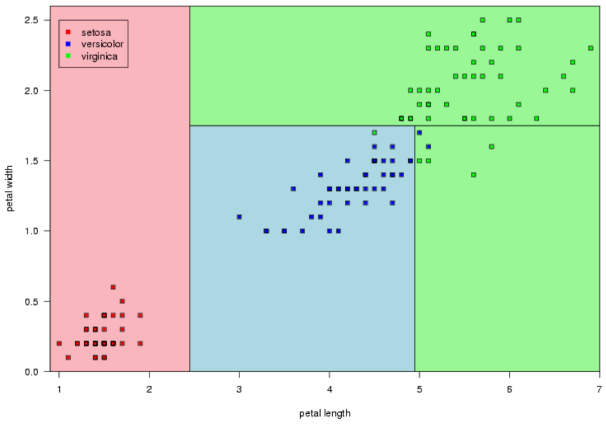
问题一：基于鸢尾花数据集的决策树分类方法

问题一中基于鸢尾花数据集的决策树分类中，首先对鸢尾花数据集中四个特征、三个类别进行可视化，直观的看出可以进行分类算法的原因；之后针对评价指标进行确定，实现了利用 5 折交叉验证计算决策树的精度，并可视化展示混淆矩阵来得到最容易出错的是种类 2 和种类 1 的鸢尾花；最后分析不同决策树最大深度对于结果的影响。从结果中可以看出当树的深度大于等于 5 之后的过拟合现象较为严重，因此综合决策树精度和防止过拟合选取 max_depth=4 较为合理，最终平均精度为 0.964.

鸢尾花数据集中主要包含 3 类共 150 个样本，每一类各 50 数据，每一条记录都有 4 项特征:[petal length，sepal width，petal length，petal width]，可以通过这四个特征来预测鸢尾花属于:[iris_setosa,iris_versicolour,iris_virginica]中的哪一个品种。

1.1 理论分析：

决策树是一个通过训练的数据来搭建起的树结构模型，根节点中存储这所有数据集和特征集，当前节点在每个分支是该节点在相应的特征值上的表现，而叶子结点存放的结果则是决策结果。通过这个模型，我们可以高效的对于未知的数据进行归纳分类，每次使用决策树时，是将测试样本从根节点开始，选择特征分支一直向下至到达叶子节点，然后得到叶子节点的分类结果。



图表 1：决策树结果举例

由上述分析可以看出决策树是直观的，也是容易解释的，属于可以理解的白盒模型，但是也由于决策树对训练数据并不作出假设，如果不加以限制，树的结果将跟随训练集变化，严密拟合，并且很可能过拟合，属于非参数模型，指的是在训练之前没有确定参数的数量，导致模型结构自由而紧密地贴近数据。解决方法主要分为约束训练模型或者采用剪枝的方法。

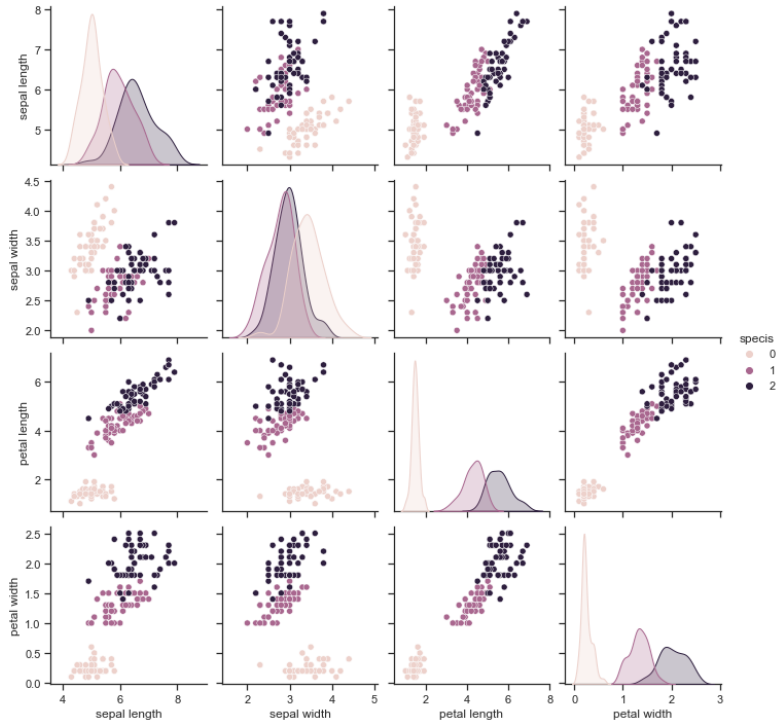
1.2 数据可视化：

首先对数据集进行分析，打印出数据集中的前五行可以大致看出数据集的基础结果，说明四种特征都是数值类型：

```
array([[4.9, 3. , 1.4, 0.2],
       [4.7, 3.2, 1.3, 0.2],
       [4.6, 3.1, 1.5, 0.2],
       [5. , 3.6, 1.4, 0.2]])
```

图表 2：鸢尾花数据集前五行特征

之后分析这四种特征对于在不同类别下的分布，这里可以用 sns.pairplot()函数来实现：



图表 3：鸢尾花四种特征在不同类别下的分布

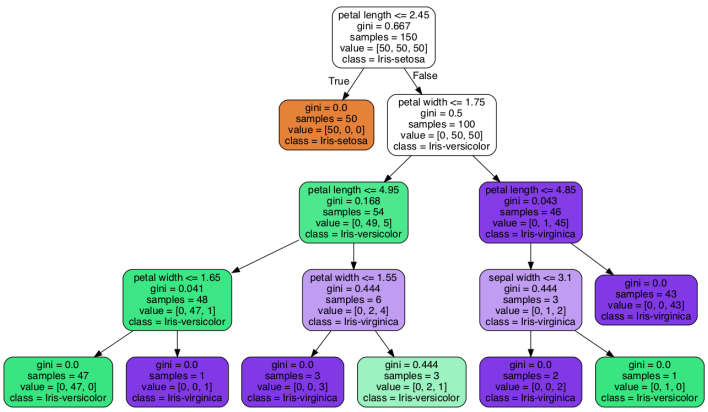
图表 3 中对角线部分主要显示了不同类别下该特征的直方图分布情况，在其余部分主要显示不同特征之间的关系以及不同类型鸢尾花中特征组合的情况，从图表观察中我们可以初步观察到：

1. 四种特征在相同类别下不一定呈现线形相关，以 speal length—petal width 在类别 2 中的走势对比，可以看出两者并没有严格的相关性，因此利用线形分类器并不合适。
2. 从特征的直方图分布中可以看出 sepal length、petal length、petal width 三种特征下三种类别区分分布分散，因此可以进行分类
3. 从两两特征中的散点图中可以看出，三种类别的区分度较好，容易区分

因此可以使用决策树算法进行分类。

1.3 评价指标：

选取 max_depth=4 对数据集进行拟合，得到最终的分类树的形状为：



图表 4：决策树分类结果

由于原始数据集中数据是均匀分布的，属于无偏数据集。因此可以尝试使用交叉验证的方式来评估该决策树模型，利用 cross_val_score()函数来计算对应的 K-folds 的训练精度，k-folds 的是将训练集分解成 K 个折叠，然后每次保存其中 1 个折叠作为预测，剩余的折叠作为训练

图表 5：k-folds 流程展示

最终输出的准确率的结果为[0.966,0.966,0.9,1,1],均大于 90%，平均准确率为 0.964，说明决策树的精度较好。

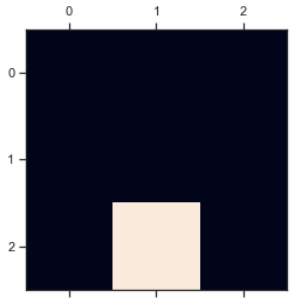
但是由于分类器本身的限制，精度在有偏数据集中很难表征分类器的精度状况，以上述为例子，如果为随机的状况，决策树的正确率至少为 33%以上，因此可以采用混淆矩阵的方式来评估分类器的性能。其构成如图：

		Condition		
	Total population	Condition positive	Condition negative	
Test outcome	Test outcome positive	**True positive**	**False positive** Type I error	Positive predictive value (**PPV**, Precision) = Σ True negative/ Σ Test outcome positive
	Test outcome negative	**False negative** Type II error	**True negative**	False omission rate (**FOR**) = Σ False negative/ Σ Test outcome negative
Positive likelihood ratio **LR+**=TPR/FPR		True positive rate (**TPR**, sensitivity) = Σ True positive/ Σ Condition positive	False positive rate (**FPR**, Fall-out) = Σ False positive/ Σ Condition negative	Accuracy (ACC) = Σ True positive + Σ True negative/ Σ Total population
Negative likelihood ratio **LR-**=FNR/TNR		False negative rate (**FNR**) = Σ False negative/ Σ Condition positive	True negative rate (**TNR**, Specificity) = Σ True negative/ Σ Condition negative	

				False discovery rate (**FDR**) = Σ False positive/ Σ Test outcome positive
				Negative predictive (**NPV**) = Σ True negative/ Σ Test outcome negative

图表 6：混淆矩阵的构成

通过将混淆矩阵中心线上的元素归 0 之后，可以用可视化的方式来展示分类器容易混淆哪些类别：



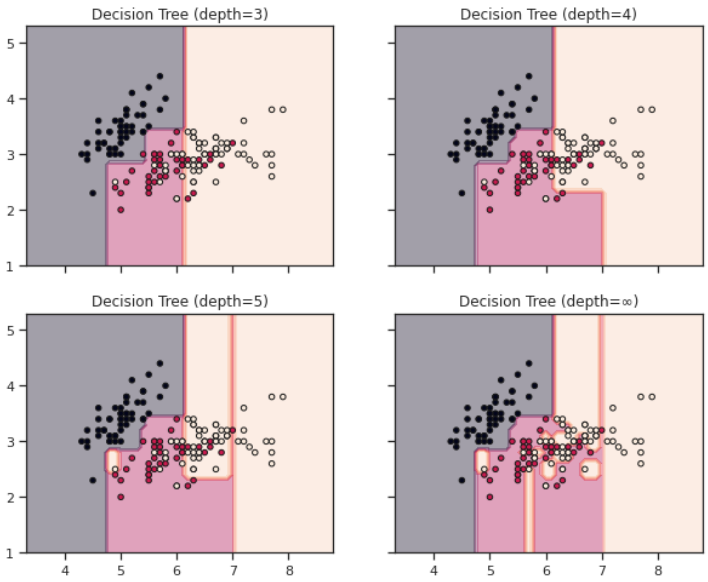
图表 7：鸢尾花--决策树混淆矩阵

由图表 7 可以看出，分类器整体的性能较好，其中出现的错误可能是将鸢尾花类别 1 和鸢尾花类别 2 之间判断错误。

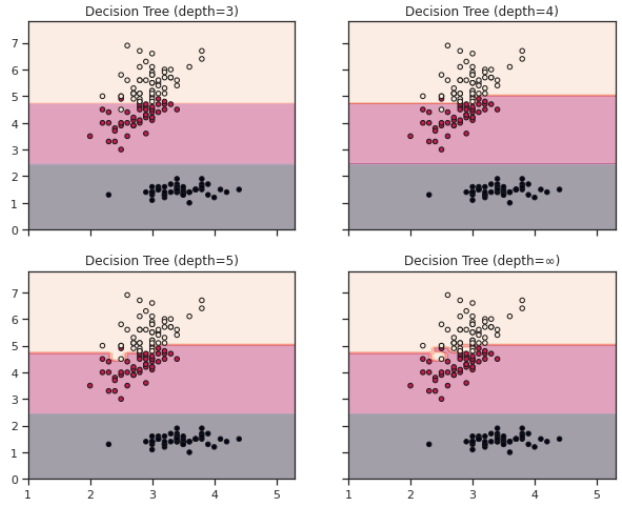
1.4 分析不同树深度的选择：

在之前决策树的分类算法的分析中也可以看出对应决策树分类中如果不限制树的增长，决策树的生成可能会对训练集产生过拟合现象，判断决策树过拟合的方式可以采用绘制出决策树边界来观察，如果决策树的边界非常不平滑，可以认为出现了过拟合的现象，这里主要对比两两特征情况下不同网络深度对于决策树结果的影响：（代码来自官网）：

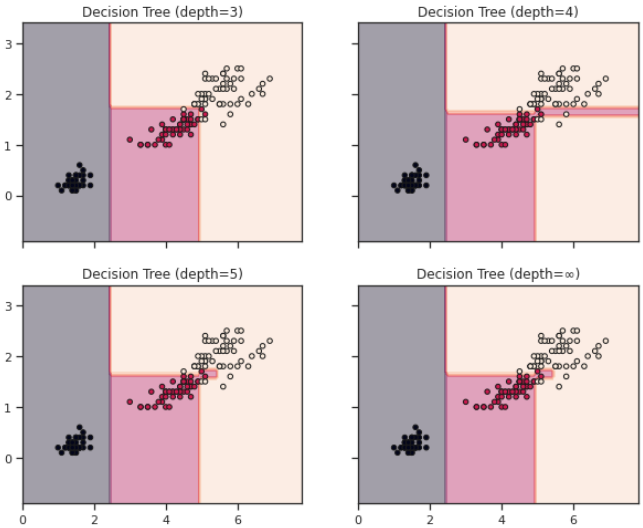
由图表 8 ~ 图表 12 可以看出，决策树的最大网络深度越大，其出现过拟合的情况就会越多，在本次鸢尾花数据集中，当 max_depth 大于等于 5 时候出现过拟合的情况较多，不适合用于决策树分类，同时在深度 3 和深度 4 的对比中可以看出，虽然 max_depth=4 也会出现过拟合现象（如图 10），但是大部分情况下其泛化性能较好，精度优于 max_depth=4,因此采用决策树最大深度为 4 可适合本次作业中参数的选取。



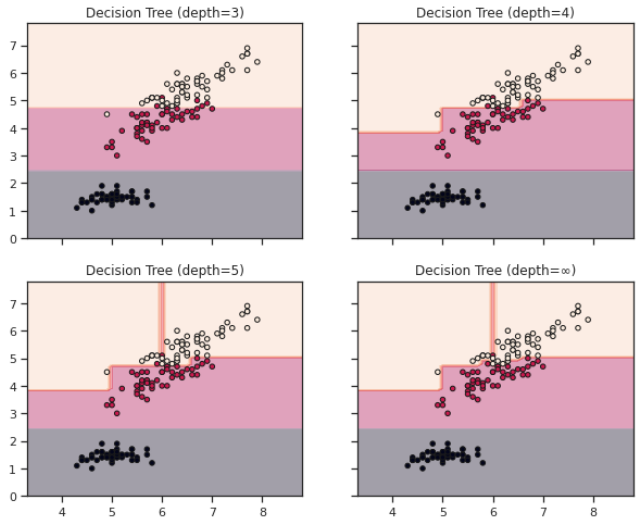
图表 8：特征 1 和特征 2 下不同网络深度决策树变化



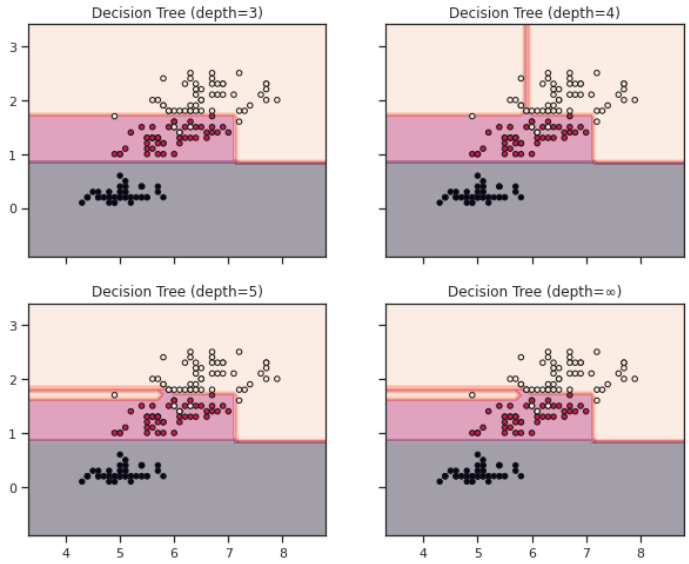
图表 9：特征 2 和特征 3 下不同网络深度决策树变化



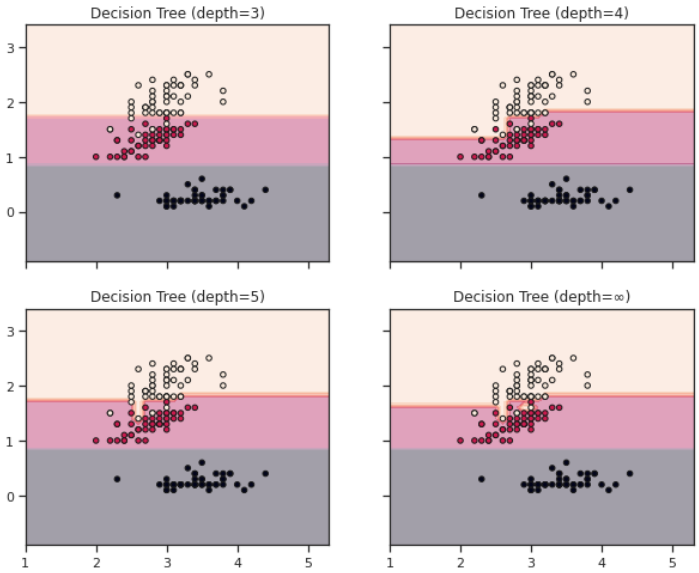
图表 10：特征 3 和特征 4 下不同网络深度决策树变化



图表 11：特征 1 和特征 3 下不同网络深度决策树变化



图表 12：特征 1 和特征 4 下不同网络深度决策树变化



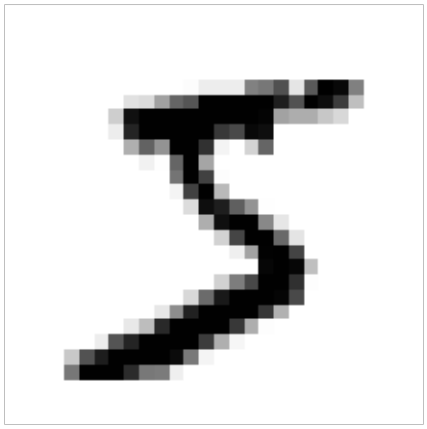
图表 13：特征 2 和特征 4 下不同网络深度决策树变化

问题二：基于神经网络的 MNIST 数据集识别

问题二中对于 MNIST 数据集的分类中，首先观察数据集中数据类型，初步的展示出数据以方便下一步操作；在可视化可以看出数据集中大部分为空白区域，因此可以采用主成分分析（PCA）的方式来对数据集进行预处理，来在保证数据集信息量的情况下降低数据集维度，之后对比处理前后网络的精度，97.56%和 97.22%，精度较为相似；之后对于该分类神经网络进行混淆矩阵的结果评价，其中看出容易混淆 4 和 9、3 和 5；同时得出上述简单的神经网络对于像素的移位和旋转具有敏感性，因此构建简单的卷积神经网络来进行识别，最终准确率达到 99.27%

2.1 数据可视化：

MNIST 数据集是一组由美国高中生和人口调查局员工手写的 70，000 个数字的图片，每一张图片都用其代表的数字标记。其前 6w 张图片为训练集，后 1w 张为测试集，首先可以考虑将特征 X 和标记 y 可视化



图表 14：训练集第一个特征值

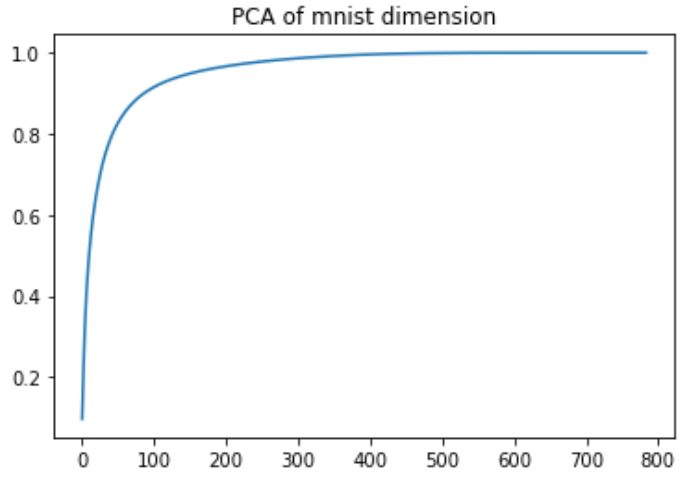
观察数据集可以看出，特征向量 X 为 28*28，共 784 个参数的二维矩阵，但是由于本次训练中要求使用神经网络对数据集进行分类，其训练过程中出现成千上万甚至数百万的特征，所有的这些特征参数不仅会让训练变得缓慢，还会让超参数调节过程中寻找更好的解决方案变的更加困难，这个问题通常称为维度的诅咒。

2.2 数据降维对于神经网络训练结果影响

在上述的训练集可视化的展示中可以看出，在 MNIST 数据集中，图像边界上的像素几乎都是白色，因此可以从训练集中压缩这些像素而不会丢失太多信息，常见的数据降维的方法有投影和流形学习，常见的数据维度技术有 PCA、kernel PCA 和 LLE，本次作业中选取 PCA 作为降维数据降维技术来观察数据降维在 MNIST 数据集神经网络识别中的结果。

主成分分析（PCA，principal component analysis）利用正交变换将一系列可能线性相关的变量转换成一组线性不相关的新变量，称为主成分，从而利用新变量在更小的维度下展示数据的特征，在空间上，PCA 可以理解为将原始数据投射到一个新的坐标系统，这种投射方式很多，为了最大限度保留对原始数据的解释，一般会利用最大方差理论或最小损失理论，是的第一主成分有最大方差或者变异系数，之后的每一个主成分都和前面的主成分正交，且有着仅次于前一主成分的最大方差，从而完全去冗余操作。

为了选择正确的主成分分析之后的维度，可以采用 explained_variance_ratio_变量来获得，该比率表示每个成分的数据集方差比率，因此可以选择相加足够大的方差部分（0.95）降低数据的额外信息，并绘制出可解释方差和降维之后的关系：



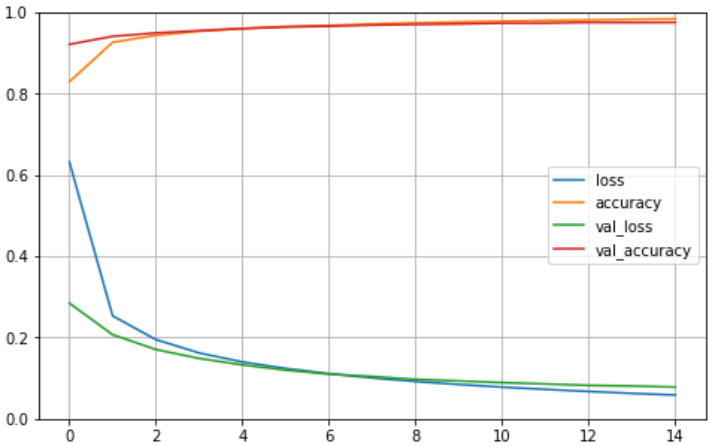
图表 15：MNIST 数据集主成分分析维度和最大方差关系

之后以 95%选取最大方差的作为方差节点，设计神经网络结构：

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
module_wrapper_4 (ModuleWrap (None, 154)               0
_________________________________________________________________
module_wrapper_5 (ModuleWrap (None, 300)               46500
_________________________________________________________________
module_wrapper_6 (ModuleWrap (None, 100)               30100
_________________________________________________________________
module_wrapper_7 (ModuleWrap (None, 10)                1010
=================================================================
Total params: 77,610
Trainable params: 77,610
Non-trainable params: 0
_________________________________________________________________
None
```

图表 16：MNIST-PCA 对应网络结果

每次迭代的训练时间为 7s，最终的网络精度为 97.56%

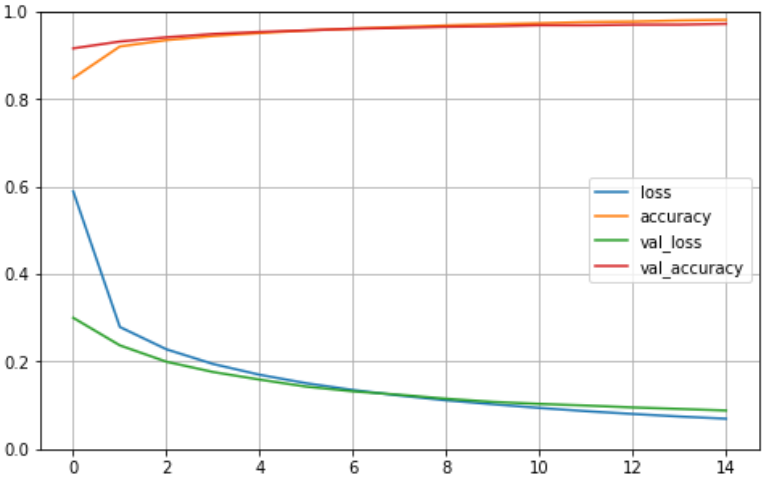


图表 17：MNIST-PCA 训练曲线

若采用未经过主成分分析来进行网络的训练，其中每轮训练的市场为 8s，最终网络的精度为 97.12%

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
module_wrapper_8 (ModuleWrap (None, 784)               0
_________________________________________________________________
module_wrapper_9 (ModuleWrap (None, 300)               235500
_________________________________________________________________
module_wrapper_10 (ModuleWra (None, 100)               30100
_________________________________________________________________
module_wrapper_11 (ModuleWra (None, 10)                1010
=================================================================
Total params: 266,610
Trainable params: 266,610
Non-trainable params: 0
_________________________________________________________________
None
```

图表 18：MNIST 原始数据集对应的网络结构

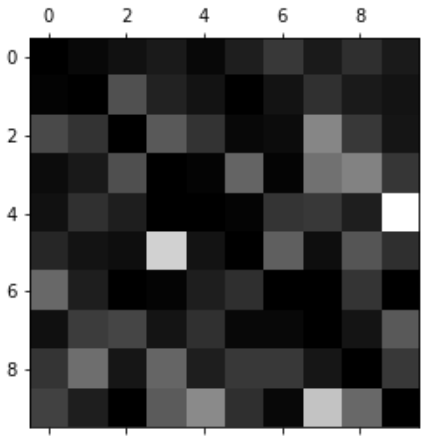


图表 19：MNIST 网络训练结果

总结：对比两者可以看出，网络结构中数据经过 PCA 的网络参数为 77610，未经过 PCA 处理的网络参数为 266610, parameters-PCA ;parameters=1:4.40 ;网络的训练时间提升 12.5%，速度得到明显的提升，网络的精度没有出现明显的下降。

2.3 神经网络识别结果分析

以图表 6 中混淆矩阵的定义来用相同的恶办啊得到对应的可视化混淆矩阵



图表 20：MNIST 数据集-神经网络分类混淆矩阵

由混淆矩阵可以看出在分类过程中数字 3 和数字 5、数字 9 和数字 4 容易发生混淆，因此可以挑选对应的 25 张识别数字 1 正确、25 张识别数字 2 正确、25 识别数字 1 为数字 2、25 识别数字 2 为数字 1，

图表 21：MNIST 识别 4 和 9 错误矩阵

图表 22：MNIST 识别 3 和 5 错误矩阵

从图表 21 和图表 22 中可以看出，MNIST 的容易混淆的矩阵可以看出其中部分肉眼看起来也会出现失误，错误比较明显。但是在其他部分中可以看出，某些错误是肉眼可以直观分辨出来而神经网络没有识别，其中的原因是因为该基础的神经网络是针对像素值的加权模型，因此对于相邻的像素值如果模式类似的话，神经网络模型会非常容易混淆，比如从 4 和 9 从像素的角度观看是否会开闭，其中主要出现的问题是像素是否产生移位或者旋转，而导致神

经网络容易将两者混淆。

2.4 利用卷积神经网络改善

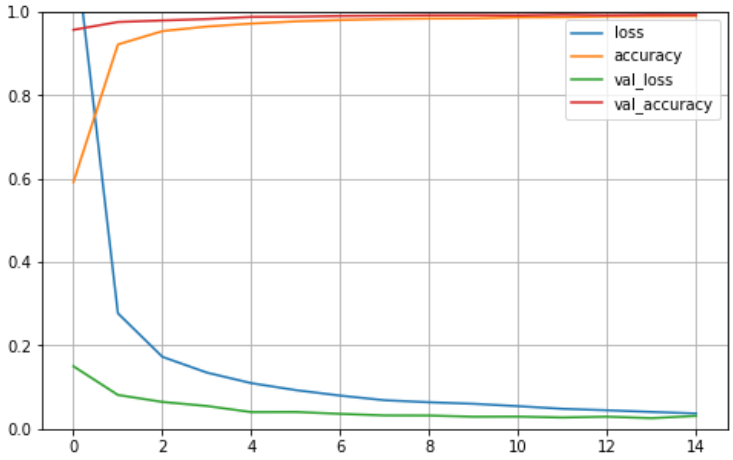
卷积神经网络（CNN）起源于对大脑的视觉皮层的研究，从 20 世纪 80 年代被用于图像识别，随着计算机计算能力的提高、可用于训练数据数量的增加、以及深度网络训练技巧的增加，让 CNN 可以实现较为复杂的视觉任务。卷积层负责提取图像中的局部特征；池化层用来大幅降低参数量级（降维），全连接层类似传统神经网络的部分用来输出想要的结果。

网络的层数结构：

```
Model: "sequential_18"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_75 (Conv2D)           (None, 28, 28, 64)        3200
_________________________________________________________________
max_pooling2d_45 (MaxPooling (None, 14, 14, 64)        0
_________________________________________________________________
conv2d_76 (Conv2D)           (None, 14, 14, 128)       73856
_________________________________________________________________
conv2d_77 (Conv2D)           (None, 14, 14, 128)       147584
_________________________________________________________________
max_pooling2d_46 (MaxPooling (None, 7, 7, 128)         0
_________________________________________________________________
conv2d_78 (Conv2D)           (None, 7, 7, 256)         295168
_________________________________________________________________
conv2d_79 (Conv2D)           (None, 7, 7, 256)         590080
_________________________________________________________________
max_pooling2d_47 (MaxPooling (None, 3, 3, 256)         0
_________________________________________________________________
flatten_15 (Flatten)         (None, 2304)              0
_________________________________________________________________
dense_45 (Dense)             (None, 128)               295040
_________________________________________________________________
dropout_30 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_46 (Dense)             (None, 64)                8256
_________________________________________________________________
dropout_31 (Dropout)         (None, 64)                0
_________________________________________________________________
dense_47 (Dense)             (None, 10)                650
=================================================================
Total params: 1,413,834
Trainable params: 1,413,834
Non-trainable params: 0
_________________________________________________________________
None
```

图表 23：卷积神经网络结构

最终的网络参数 1413834，网络精度为 99.06%



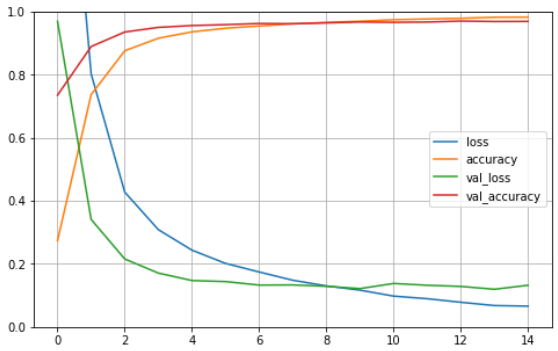
图表 24：卷积神经网络训练过程

利用 PCA 对 MNIST 数据集处理后（由于卷积神经网络需要输入图像，此处的维度选择

13*13=169)，后重新构建网络训练，最终的训练精度为 96.94%，其中网络的过拟合程度降低

```
Model: "sequential_21"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_90 (Conv2D)           (None, 13, 13, 64)        3200
_________________________________________________________________
max_pooling2d_54 (MaxPooling (None, 6, 6, 64)          0
_________________________________________________________________
conv2d_91 (Conv2D)           (None, 6, 6, 128)         73856
_________________________________________________________________
conv2d_92 (Conv2D)           (None, 6, 6, 128)         147584
_________________________________________________________________
max_pooling2d_55 (MaxPooling (None, 3, 3, 128)         0
_________________________________________________________________
conv2d_93 (Conv2D)           (None, 3, 3, 256)         295168
_________________________________________________________________
conv2d_94 (Conv2D)           (None, 3, 3, 256)         590080
_________________________________________________________________
max_pooling2d_56 (MaxPooling (None, 1, 1, 256)         0
_________________________________________________________________
flatten_18 (Flatten)         (None, 256)               0
_________________________________________________________________
dense_54 (Dense)             (None, 128)               32896
_________________________________________________________________
dropout_36 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_55 (Dense)             (None, 64)                8256
_________________________________________________________________
dropout_37 (Dropout)         (None, 64)                0
_________________________________________________________________
dense_56 (Dense)             (None, 10)                650
=================================================================
Total params: 1,151,690
Trainable params: 1,151,690
Non-trainable params: 0
_________________________________________________________________
None
```

图表 25：MNIST_PCA_CNN 构建的网络



图表 26：MNIST_PCA_CNN 网络训练结果

两者网络参数为：1413843 与 1151690，参数明显降低，网络的训练速度从每轮训练 40s 降低 20s，但是经过 PCA 得到的网络容易出现过拟合现象，同时网络的精度降低

对上述所有分析的结果总结：

	NN	PCA-NN	CNN	PCA-CNN
每轮训练时间/s	8	7	53	20
参数数目	26610	77610	1413834	1151690
测试集准确率	0.9813	0.9842	0.9906	0.9845
验证集准确率	0.9722	0.9756	0.9927	0.9694

Reference：

https://github.com/chenxia31/handson-ml2

https://github.com/chenxia31/Machine-Learning