王宇睿 2015201906

人工智能课程总结

第一周 About the class

- 课堂笔记

第一周的内容主要是课程安排及介绍，在课堂上了解了人工智能的历史、方法、借助的工具以及一些应用。

- 上机练习

本周未安排上机。

- 课后工作

属于了解阶段，暂未进行课后练习。

第二周 Arduino Basics

- 课堂笔记

课程大作业AiCar需要使用Arduino板来控制小车。Arduino板的硬件部分是可以用来做电路连接的Arduino电路板，还有Arduino IDE，是Arduino板的程序开发环境。只要在IDE中编写程序代码，将程序上传到Arduino电路板后，程序会控制Arduino电路板。
Arduino板有很多端口，可以连接蓝牙、传感器等硬件，并使用程序逻辑控制小车的行为。
Arduino编程语言和C语言很类似，一些简单实用的命令如：

Command	Description
`pinMode(n,INPUT)`	Set pin *n* to act as an input. One-time command at top of program.
`pinMode(n,OUTPUT)`	Set pin *n* to act as an output
`digitalWrite(n,HIGH)`	Set pin *n* to 5V
`digitalWrite(n,LOW)`	Set pin *n* to 0V
`delay(x)`	Pause program for *x* millisec, *x* = 0 to 65,535
`tone(n,f,d)`	Play tone of frequency *f* Hz for *d* millisec on speaker attached to pin *n*
`for()`	Loop. Example: for (i=0;i<3;i++){} Do the instructions enclosed by {} three times
`if (expr) {}`	Conditional branch. If *expr* true, do instructions enclosed by {}
`while (expr) {}`	While *expr* is true, repeat instructions in {} indefinitely

- 上机练习

本周未安排上机。

- 课后工作

安装Arduino IDE，运行ppt中Arduino的示例代码。

第三周 Aicar Building

- 课堂笔记

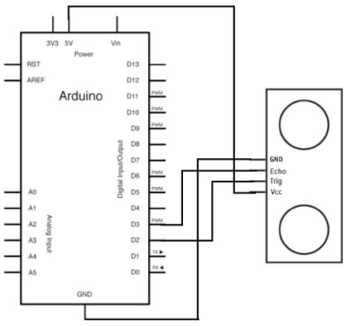
硬件：Arduino UNO、BT06 Bluetooth、HC-SR04 Ultrasonic Distance Measuring、L293D Motor Drive Shield、2WD Smart Robot Car Chassis Kit
软件：DroidCam for Android、DroidCam Client for Win/Linux、Arduino IDE (C/C++)、Matlab、Python
小车前进forward()：注意由于购买的马达可能动力不同、小车车轮也不完全相同，因此设置小车前进速度时，左轮和右轮的速度不应该设置为一样的，需要通过测试调整RLRatio。
超声波测距：编写getdistance()获得单位时间内探测器得到的间隔距离，当距离小于40cm时，随机右转。
蓝牙连接：可借助Matlab或python。
使用手机相机获得图像：可借助Matlab，同时可使用imresize和imrotate处理图像。也可以借助python的OpenCV。

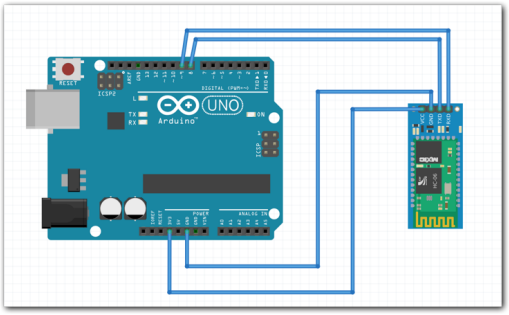
- 上机练习

本周未安排上机。

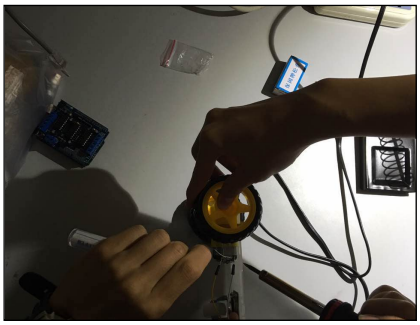
- 课后工作

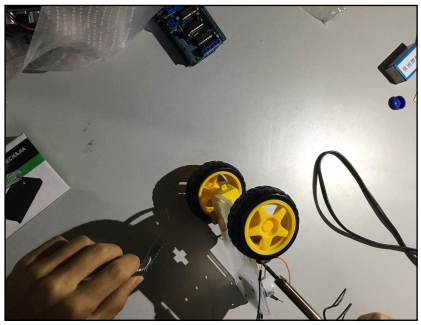
AiCar制作第一阶段，安装小车，连接电机驱动、超声波、蓝牙模块。电机驱动使用L293D模块，用于驱动两个电机。超声波和蓝牙模块与Arduino扩展板的连接示意图分别为：

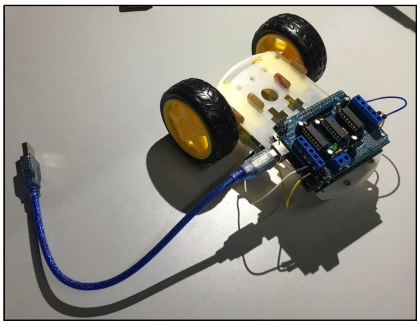




此阶段小组完成小车的基本搭建，利用并修改ppt中的代码，实现小车的随机行走，同时连接完成蓝牙模块，可以通过手机发送数字来控制小车，完成超声波避障。







第四周 Machine Learning Landscape

- 课堂笔记

根据训练是否受监督，机器学习可分为监督学习、无监督学习、强化学习以及半监督学习。监督学习算法：KNN、线性回归、逻辑回归、SVM、决策树、随机森林、神经网络；无监督学习算法：聚类：k-Means、HCA、EM算法；降维：主成分分析PCA、局部线性嵌入LLE、t-SNE。
机器学习中的挑战主要是bad data & bad algorithm。Bad data主要是训练数据不足、训练数据不具有代表性、无关特征（garbage in, garbage out）；Bad algorithm主要特征是训练数据过拟合（超参数的选择）、训练数据低度拟合。
Docker使用过程：安装Docker，相当于构造虚拟机，模拟在Linux中跑，进入docker后，打docker可以看到各种使用命令，使用镜像images（区分CPU、GPU），打docker run+镜像名可以启动。建立端口映射，再运行docker，提示网址（把localhost改成之前提示的IP），即可使用notebook运行ipynb。

- 上机练习

本周未安排上机。

- 课后工作

由于是macOS系统，类似Linux系统，且已经安装好需要使用的python库，因此未使用docker；安装Jupyter Notebook，运行书中给的第一章代码。

① 从网上提供开源数据集gdp_per_capita.csv下载数据并尝试使用sort_values函数根据不同列的值排列数据。
② 用简单的线性模型对gdp_per_capita和life_satisfaction进行拟合，绘制图像，并观察过拟合模型和根据所有数据、部分数据进行线性拟合的图像。

第五周 End-to-End Machine Learning Project

- 课程笔记

先进行了课程大作业第一次小组展示并投票。
之后以加州1990年收集到房价为数据集，使用机器学习算法预测最近加州的房价水平，了解End-to-End Machine Learning Project的构建过程。
分析问题：由于数据集是被标注的数据，因此该问题是典型的监督学习；同时，这是一个多元回归问题，因为我们需要多个特征来进行预测；由于没有连续的数据，数据集较小，batch learning就可以完成任务。
选择模型评价指标：RMSE、MAE等。
获得数据：确定数据集下载地址。
展示数据：可以借助matplotlib进行绘图，比如用hist绘制频数直方图。
查看数据相关性：使用pandas中的scatter_matrix()，它会描绘出数据间的两两关系。

特征结合：注意到一些原特征对于预测房价是没什么用的，如total_rooms与total_bedrooms，很容易想到与其相关且与房价有关的属性应该是rooms_per_household，bedrooms_per_room还有population_per_household。所以利用如total_rooms，total_bedrooms，households与population等特征结合来生成新的特征。编写CombinedAttributesAdder进行特征结合。
建立测试集：采用分层取样。
特征归一化处理：min-max scaling (normalization) and standardization。
pipline：由于数据需要进行多步处理，为保证顺序处理，可以构造pipline。pipline首先使用Imputer来处理缺失值，再用CombinedAttributesAdder进行特征结合，最后进行特征归一化处理。
模型评估：使用交叉验证以评估模型表现。可以看到决策树模型的表现甚至还不如线性模型，决策树模型有过拟合的问题。还可以选择另一个学习模型进行尝试：随机森林。
模型调优：使用网格搜索GridSearchCV，只需要指定各参数的取值，然后函数会自动随模型进行训练、评估并选出最优参数。

- 上机练习

本周未安排上机。

- 课后工作

运行书中第二章代码。

第六周 Classification

- 课堂笔记

以MNIST数据集手写体识别为例，首先介绍了二分类问题：判断一个输入图片中的数字是不是5。
选择分类器SGD **(Stochastic Gradient Descent)**随机梯度下降：该分类方法通过每个样本来迭代更新一次，如果样本量很大的情况，比如几十万，那么可能只用其中几万条或者几千条的样本，就已经将参数迭代到最优解，训练速度快，适合应用于大规模数据集。可以直接调用scikit-learn中的SGDClassifier进行使用。
在衡量分类器的表现时，可以K折交叉验证（**K-fold cross validation**）：把样本集分成k份，分别使用其中的(k-1)份作为训练集，剩下的1份作为交叉验证集，最后取最后的平均误差，来评估模型。
但是要注意，很多时候仅仅使用accuracy并不是衡量分类器表现的最好指标，更好的方法是使用混淆矩阵进行分析。
混淆矩阵（**confusion matrix**）：每一列代表预测值，每一行代表的是实际的类别。混淆矩阵要统计正确预测到的负例的数量，把负例预测成正例的数量，把正例预测成负例的数量，以及正确预测到的正例的数量，这样就可以用于衡量分类准确率、正例覆盖率、正例命中率等。可以直接使用scikit-learn中提供的confusion_matrix()获得混淆矩阵。

接下来介绍多分类问题，识别0-9共十个手写数字。一种方法是构造十个二分类器，在判断测试数据时，输出最高得分的分类，这被称为**one-versus-all** (OvA) strategy；或者也可以训练45个（C_n^2）二分类器，用于判断一对数字，如0和1，0和2等，这叫做**one-versus-one** (OvO) strategy。这两种方法在sklearn中都有实现，分别为OneVsOneClassifier和OneVsRestClassifier。

- 上机练习

使用书中提供的代码，在MNIST数据集上进行手写体识别分类。

注意在跑到第31个cell时，precision_recall_curve()语句会报错，提示ValueError: bad input shape，这是因为电脑系统原因，我没有安装docker，因此python中安装的sklearn版本较高，和代码中要求的不匹配，造成了函数定义的细小差别，将sklearn降低版本安装即可。

书中提供的代码除了包括课堂的内容，还额外补充了KNN（**K Neighbors Classifier**）分类器的使用。 KNN是通过测量不同特征值之间的距离进行分类，思路是：如果一个样本在特征空间中的k个最相似(即特征空间中最邻近)的样本中的大多数属于某一个类别，则该样本也属于这个类别。在KNN算法中，所选择的邻居都是已经正确分类的对象。该方法在定类决策上只依据最邻近的一个或者几个样本的类别来决定待分样本所属的类别。

代码中有个小trick，通过上下左右不同方向移动数据集的方式，可以扩大训练集，从原来的70000扩大到300000，accuracy也有一点提升。总的来说，KNN算法用于MNIST数据集手写体识别准确率能达到97%以上，比SGD好一些。

- 课后工作

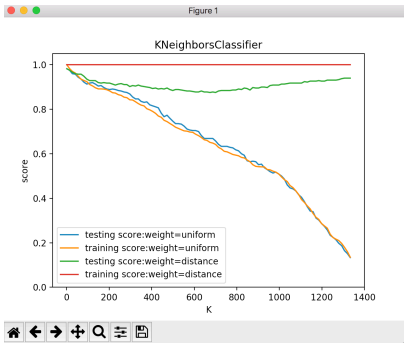
阅读学习《python大战机器学习》的K邻近章节，使用 scikit-learn 自带的手写识别数据集 **Digit Dataset**（由于MNIST数据集太大，速度较慢）进行手写体识别分类， 同时测试 KNN算法中 n_neighbors、weights 和p 参数的影响。

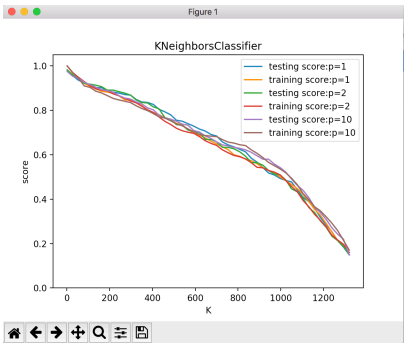
使用交叉验证，测试集与训练集的比例为1:3，使用KNN分类器，在训练集上最好的accuracy为0.991091，在测试集上最好的accuracy为0.980000。

测试 KNN 中 n_neighbors 和 weights 参数的影响，使用numpy中的linspace方法创建n_neighbors（k值）的等差数列（数据集大小为1797*64，因此k值只取到1400），并选取weights为uniform、distance两种，绘制不同 weights下， accuracy 随 n_neighbors 的曲线。

可以看到，weights=uniform时得分随K值的增加而下降，而weights=distance时，K值的影响不算太大。weights参数是在进行分类判断时给最近邻附上的加权，uniform是等权加权，distance是按照距离的倒数进行加权。当K值较小意味着只有与输入实例较近的训练实例才会对预测结果起作用，但当K值较大时，学习的近似误差增大，这时与输入实例较远的训练实例也会对预测起作用，使预测发生错误，因此weights=uniform时得分受K值影响很大。

测试 KNN 中 n_neighbors 和 p 参数的影响，绘制不同p值下， accuracy 随 n_neighbors 的曲线。





p参数用于衡量两个实例点相似程度的反映，距离公式可表示为：

$$d((x_1,\ldots,x_n),(y_1,\ldots,y_n)) = \left(\sum_{i=1}^{n}|x_i - y_i|^p\right)^{1/p}$$

当p=2时，表示欧氏距离；当p=1时，表示曼哈顿距离；当p很大时，距离取各维度距离中的最大值。
可以看到在进行分类时，KNN分类器的参数选择对accuracy的影响很大。在进行参数选择时，可以使用GridSearch网格搜索。

第七周 Training Models

- 课堂笔记

使用数据对模型进行训练，寻找模型的最优参数。
线性模型：线性模型最简单，有直接的数学计算公式计算使得cost function最小的参数：$\hat{\theta} = (X^T \cdot X)^{-1} \cdot X^T \cdot y$。使用numpy库中提供的rand函数生成在线性模型 $y = 4 + 3X$ 周围的随机数（Gaussian noise），并用numpy提供的Linear Algebra module计算矩阵转置和矩阵乘法，得到公式法计算出的theta_best，可以用这个 ϑ 来进行函数值预测。而sklearn库中提供了现成的方法进行线性回归，得到的结果与公式法得到的一致。线性模型一旦训练好，进行预测时非常快。
可以用来训练线性模型的方法有：
梯度下降：一开始进行随机初始化，逐步优化，目标是减小cost function，直到收敛。对于梯度下降算法，一个重要的参数就是下降的步长，对应的超参数是学习率。学习率如果设置的太小，可能会迭代太多次都得不到收敛的结果，花费太长时间；如果设置的太大，可能会越过最佳点，得不到最优解。同时，梯度下降法可能发现的是局部最优，而不是全局最优。不过MSE cost function是凸函数，这使得梯度下降法发现的一定是全局最优。
批量梯度下降：对cost function求偏导，按照梯度的负方向下降，下降的步长为学习率，在每次更新参数时都使用所有的样本来进行更新。
随机梯度下降：由于批量梯度下降法在更新每一个参数时，需要所有的训练样本，所以训练过程会随着样本数量的加大而变得缓慢，因此出现了随机梯度下降法来解决这一弊端。随机梯度下降法利用每个样本的损失函数对 ϑ 求偏导，得到对应的梯度来更新 ϑ，训练速度较快。

学习曲线：如果训练好的模型在训练集上表现很好，但使用交叉验证的测试集上表现很差，那么说明数据出现了过拟合的情况；如果模型在训练集和测试集上变现都很差，则说明数据出现了欠拟合的情况。
偏差 / 方差：高偏差意味着模型在训练集上欠拟合，高方差意味着模型在训练集上过拟合。
岭回归：是一种正则化版本的线性回归，为损失函数加上一个正则化项，避免矩阵中某个元素的一个很小的变动引起最后计算结果误差很大。岭回归有超参数α，如果α为0时，就化简为线性模型，岭回归牺牲了一部分无偏差性，使得方差变小。
逻辑回归：将数据拟合到一个logistic函数。为估计概率，使用sigmoid函数，它是一个s形的曲线，取值在[0, 1]之间。使用逻辑回归可以更好的对事件发生的概率进行预测，以鸢尾花数据集为例说明逻辑回归。只基于花瓣宽度这一特征构造逻辑回归模型，用于检测Iris-Virginica种类。
Softmax回归：Softmax回归是逻辑回归的推广，逻辑回归处理二分类问题，而Softmax回归处理多分类问题，类标签y的取值大于等于2。对于给定的测试输入x，利用模型针对每一个类别j估算概率值p(y = j|x)，Softmax函数将k个可能的类别进行了累加，对于Softmax的代价函数，利用梯度下降法使的J(θ)最小。

- 课后工作

小组进行课程大作业AiCar功能完善，尝试自己构建简单的语音识别系统，识别包括“左”，“右”，“前”，“后”，“停”等控制小车前进方向的关键字，构建和识别的主要过程为：

- 数据采集及特征提取

  通过录制音频和开源数据中采集两种方式获得了用于输入大量的语音数据和其对应的文字标签。对获得的原始语音信号进行处理（由于录制时在较为安静的环境中，因此不需进行降噪处理），对语音信号进行分帧（近似认为在10-30ms内是语音信号是短时平稳的，将语音信号分割为一段一段进行分析）以及预加重（提升高频部分）等处理。

  用kaldi提取出能够反映语音信号特征的关键参数MFCC特征。

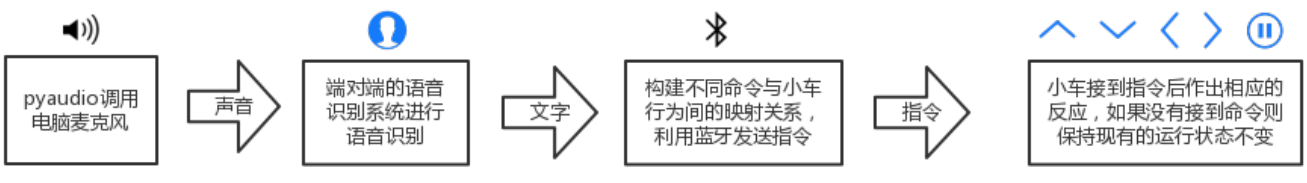
- 模型训练

  将整理好的MFCC特征和对应的类标签输入到构建的分类器中进行训练，利用交叉验证和网格搜索调整分类器的超参数，获得构建好的语音识别模型。

- 识别命令

  利用python中的pyaudio包调用电脑的麦克风，从麦克风中获取声音。每隔一定的秒数将获取到的声音文件实时保存到本地。在linux环境中调用shell脚本利用现有的工具对音频文件进行MFCC特征提取，将得到的特征结果结果以文件形式保存，然后程序从文件中读取数据，输入到语音识别模型中进行分类，得到对应的标签（左、右、前、后、停）。

然而，由于从麦克风中获得声音后进行的操作较多，速度慢，同时，自己构建的语音识别系统正确率欠佳，无法满足控制小车的实时性要求，因此目前为止，我们小组决定采用现有的模型对小车进行语音控制。
首先在电脑端利用端对端的语音识别系统（IBM Speech to Text）进行语音识别，识别出所说的内容，然后程序读取语音的内容，并根据语言的内容给小车发送相应的指令，小车接到指令后作出相应的反应，如果没有接到命令则保持现有的运行状态不变。整个语音识别过程见下图：



第八周 SVM、Decision Trees

- 课堂笔记

SVM是非常通用并且有效的机器学习模型，可以处理线性和非线性的分类问题、回归问题以及孤立点检测。SVM寻找一个分类面，使得两个点集到此平面的最小距离最大，两个点集中的边缘点到此平面的距离最大。注意SVM对于特征的范围非常敏感，因此可以使用sklearn中提供的方法进行特征缩放。SVM可分为Hard-Margin和Soft-Margin，Soft-Margin不能容忍数据集中的噪声，因此会造成过拟合的问题；而为了让Hard-Margin容忍一定的误差，在每个样本点后面加上了一个宽松条件，允许这个点违反一点误差，称为Soft-Margin。在sklearn库中实现的SVM类可以通过调整超参数C来控制允许的误差，如果C很大，对错误的惩罚越大，模型会倾向于尽量将样本分割开；如果C越小，则会有更多的违反边界的点存在，并且图中的margin就越大。对于线性不可分的数据，利用kernel将数据从一个空间转换到另一个空间，更易于分类器进行分类。常用的核函数有多项式函数，多项式核函数可以实现将低维的输入空间映射到高纬的特征空间，但是多项式核函数的参数多，当多项式的阶数比较高的时候，核矩阵的元素值将趋于无穷大或者无穷小，计算复杂度会大到无法计算。除此之外，还可以使用RBF，RBF高斯径向基函数是一种局部性强的核函数，可以将一个样本映射到一个更高维的空间内，该核函数是应用最广的一个，无论大样本还是小样本都有比较好的性能，而且其相对于多项式核函数参数要少，因此大多数情况下在不知道用什么核函数的时候，优先使用高斯核函数。

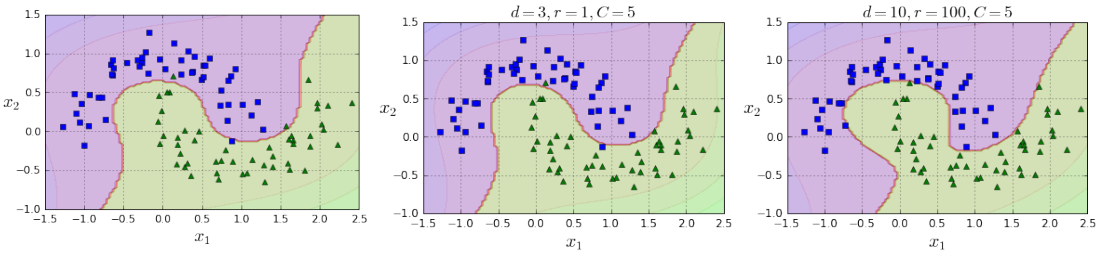
除了解决分类问题，SVM还能解决回归问题，SVM回归试图适合最多的样例，模型由误差较大的点决定。在SVM回归算法中，目的是训练出超平面，利用超平面进行预测。与分类问题一样，支持向量机的回归预测从线性到非线性转换是通过核函数，核函数就是一种映射，所以选择不同的核函数，那么结果将会形成不同的算法。

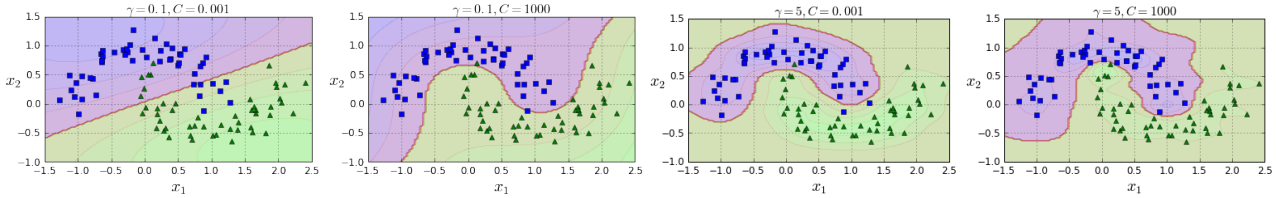
接下来，为更好地理解决策树模型，使用鸢尾花数据训练并图像化一个决策树模型。决策树的构造就是进行属性选择度量确定各个特征属性之间的拓扑结构，在某个节点处按照某一特征属性的不同划分构造不同的分支，其目标是尽量让一个分裂子集中待分类项属于同一类别。在每次需要分裂时，计算每个属性的增益率，然后选择增益率最大的属性进行分裂。在构造决策树时，通常要进行剪枝，为了处理由于数据中的噪声和离群点导致的过分拟合问题。

- 上机练习

运行提供的第五、六章代码。

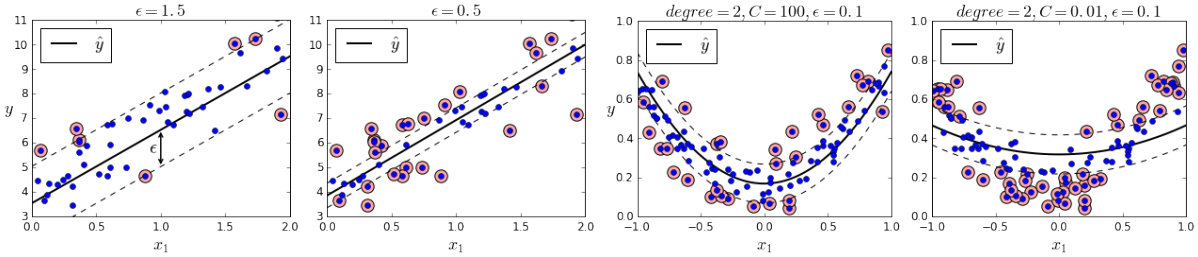
① 利用鸢尾花数据集训练并测试SVM模型。构造pipeline，生成多项式特征，多项式阶数选为3，对数据进行预处理，利用StandardScaler()将数据转为标准正态分布，并将处理好的数据送入不同的SVM分类器。首先核函数选取linear，分类器超参数C=10，选取损失函数为hinge，random_state=42。再测试多项式核函数，超参数C=5，比较degree=3、coef0=1和degree=10、coef0=100时得到的预测曲线。最后测试RBF核，分别取不同的超参数gamma和C，查看分类曲线。





② 测试Gaussian_RBF核将一个样本映射到一个更高维空间的能力，并了解使用核函数后对分类效果的提升。

③ 生成随机数据测试SVM回归模型，分别选取线性以及多项式核函数绘制预测曲线，并观察不同超参数值对预测的影响。



- 课后工作

1. 利用所学分类器，进行多模态情感识别。使用RML数据集，根据数据集视频中人物的声音、表情或肢体动作等信息预测该人物的情感类别，构建并测试不同情感识别分类器，包括SVM、Random Forest，调整超参数，提高识别准确率。

语音情感识别：

对数据集中s1~s8中所有视频提取语音特征，并将同一个人的不同视频（由于不同说话人情感差异大，故分说话人对特征进行处理）进行归一化处理。将所有语音特征和标签整理为numpy数组，数组一行为一个视频提取出的语音特征，对应的标签包括anger, disgust, fear, sad, surprise, happy。使用交叉验证对模型进行评价，比较不同超参数数值、不同分类器的影响。

构造了SVM分类器，在保持kernel=rbf，gamma为默认值和cross_validation的参数cv=10保持不变时，选取C从2的幂次方依次调节（0.25-128），得到结果当C取到4时，模型预测效果最好，最佳结果accuracy为0.90277778，平均表现accuracy为0.7625，用时为14.536531s，之后C继续变大（4-128），并不改变预测效果，但用时变长。

取gamma为默认值和cross_validation的参数cv=10，C=4保持不变时，取kernel=rbf和linear，通过实验发现Linear核已经取到较好的效果（最佳结果accuracy为0.8888889），并且用时仅为Rbf核的65%。若仅考虑精度要求，可以选择Rbf核，若有速度要求，可优先选择Linear核。

之后构造了Random Forest分类器，并考察参数max_depth以及n_estimators的影响。max_depth表示单棵树的最大深度，深度越小，计算量越小，速度越快，但深度小时精度较差，控制决策树深度目的是防过拟合。在RML数据集中，max_depth=6时表现已经不错，accuracy为0.83333333。n_estimators表示随机森林中树的数量，经过控制变量的实验可以发现树的数量越大，计算时间越长，而树的数量变化对精度的影响并不是很大。当

n_estimators=160，其余参数取默认值时，最佳结果accuracy为0.8888889。在不设置n_estimators时，随机森林分类器的速度比SVM快很多，但牺牲了精确性。
使用网格搜索GridSearchCV对SVM分类器进行参数调优，网格搜索的结果显示，当C取大于等于1的2的幂次方时，对结果不造成影响，最佳结果accuracy为0.90277778，与交叉验证结果相近。
图像情感识别：
将数据集s1~s8中的每个视频按照每秒3张的速度提取图片，使用openface对图片进行姿势检测和校准，并将图片的大小裁剪为96，共获得处理过的图片6478张。利用openface提取图片特征，获得6个标签，及特征矩阵6478*128。测试SVM的linear核、rbf核，DecisionTree，GaussianNB，DBN等分类器，并使用GridSearchCV取cv=5来寻找SVM的最佳参数。经过实验，最佳的结果是取SVM分类器，最佳参数是C=1,kernel='linear'，但预测结果的accuracy50%左右，可能是数据集并不具有普遍性。

2. 小组继续完善AiCar功能，使得小车可以自动追踪红色电烙铁。将手机固定在小车上，利用软件将手机作为网络摄像头，使得电脑可以通过相应的ip访问手机摄像头，通过这种方式，将手机拍摄到的画面实时传送到电脑。在电脑端，我们使用opencv对传回的视屏进行处理，本次实验中，我们对识别的规则进行了设定，利用红色电烙铁上的红色方形区域，使得程序能够根据颜色和形状的信息识别视频中的电烙铁，并给出电烙铁上的红色方形区域的质心在小车的视野中的坐标，根据坐标信息判断电烙铁在小车的那个方向，然后通过蓝牙串口给小车发送改变运动方向的指令，小车收到指令后会做出相应的反应，从而使小车跟踪视屏中的电烙铁。

第九周 Ensemble Learning and Random Forests

- 课堂笔记

这章介绍了很多集成学习方法，并介绍了随机森林。
投票分类器：投票分类器的原理是结合了多个不同的机器学习分类器，使用多数票或者平均预测概率，预测类标签。这类分类器对一组相同表现的模型十分有用，同时可以平衡各自的弱点。
Soft Voting：返回预测概率值的总和最大的标签，可通过参数weights指定每个分类器的权重；若权重提供了，在计算时则会按照权重计算，然后取平均；标签则为概率最高的标签。
Bagging and Pasting：获得多样分类器的一种方法是使用不同的算法，另一种方法是使用相同的算法，但是在训练数据的不同子集上进行训练。如果针对训练数据的抽样是有放回的抽样，那么这样的方法是bagging，如果针对训练数据的抽样是没有放回的抽样，那么这样的方法是pasting。相比pasting，bagging有放回的抽样虽然会略微提高偏差，但是能够提高模型的多样性，降低方差。总的来说，bagging通常能够取得更好的效果。但是，如果时间允许，可以考虑比较bagging和pasting的结果。
Out-of-Bag Evaluation：在bagging的过程中，部分训练数据可能会被多次抽样，部分训练数据可能不会被抽样。因为训练特定模型时，oob的训练数据是不被使用的，所以可以利用这些数据对当前模型进行验证，而不需要单独的验证集或进行交叉验证。可以通过这些结果的平均值衡量集成学习的结果。在sklearn里，可以通过设置oob_score=True实现。
随机森林：随机森林是决策树的集成，通常根据bagging的思想训练，max_samples通常是训练数据的规模。随机森林在训练决策树分裂结点时不是从所有特征中寻找最优的特征，而是从

部分特征中寻找最优的特征。能够提升模型多样性，从而增加偏差，减少方差，通常，训练的效果能够得到提升。

特征重要度：决策树中，重要的特征往往靠近树的根部，不重要的特征往往靠近树的底部，或者不出现在决策树结点，能够根据森林中特征的平均深度判断特征的重要程度。scikit-learn自动进行以上过程，可以通过feature_importance获得特征的重要程度。以iris数据为例，petal length（44%）和petal width（42%）是相对重要的特征，sepal length（11%）和sepal width（2%）是相对不重要的特征。

AdaBoost：当使用AdaBoost进行分类时，当前模型纠正错误的方法包括更加关注前面模型的错误，这样的过程导致新的模型越来越关注困难的训练数据。AdaBoost逐个训练模型的思想类似梯度下降。不同的是，梯度下降尝试通过寻找最优参数最小化代价函数，AdaBoost通过添加新的模型提升模型的效果。所有模型训练完成后，AdaBoost使用类似bagging的方式整合结果，只是不同的模型根据他们在带权训练数据的预测结果拥有不同的权重。

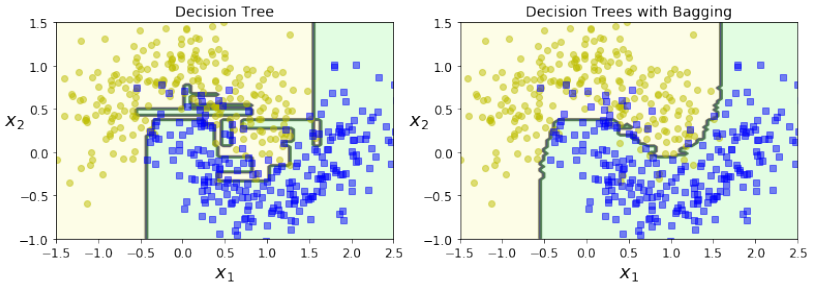
Gradient Boosting：和AdaBoost类似，Gradient Boosting逐个训练模型，当前的模型尝试纠正前面的模型的错误。但是，AdaBoost纠正错误的方法是更加关注前面模型的错误，Gradient Boosting纠正错误的方法是拟合前面模型的残差。

- 上机练习

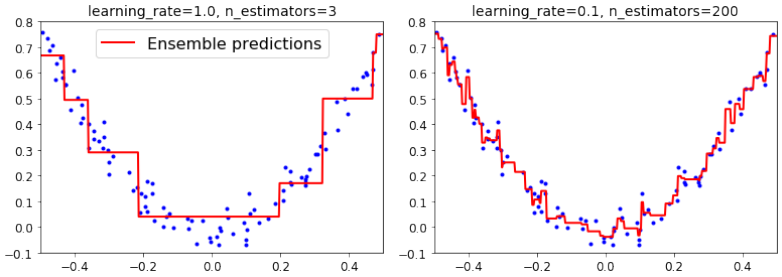
运行提供的第七章代码。

① 通过构造的数据集利用逻辑回归，随机森林和SVM训练voting classifier。注意默认情况下SVM不能够预测类别的概率，需要设置参数probability，这样的设置会降低训练速度。可以发现，hard voting classifier达到89.6%的准确率，soft voting classifier达到91.2%的准确率，均高于单独的分类器。

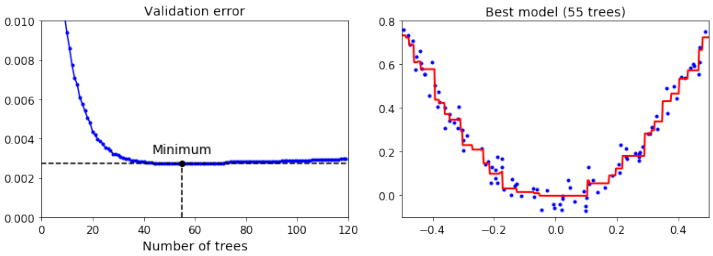
② 比较决策树的决策边界和基于决策树的BaggingClassifier的决策边界。虽然两种模型在训练数据上的结果类似，但是BaggingClassifier的决策边界更加平滑，在测试数据上的结果更加优秀，说明集成学习得到的结果具有类似的偏差和较低的方差。



③ 通过sklearn提供的GradientBoostingRegressor实现Gradient Boosting，有控制决策树训练的参数（max_depth）和控制模型集成的参数（n_estimators）。测试不同控制模型集成参数的训练结果，可以看到n_estimators=3的模型基础模型过少，存在欠拟合的问题，而n_estimators=200的模型基础模型过多，存在过拟合的问题。



④ 寻找基础模型的最佳数量，使用early stopping，在验证数据效果最优时停止训练。实现以上思想的简单方法是使用staged_predict方法，其在训练的所有阶段分别进行预测。通过staged_predict方法确定基础模型的最佳数量，分别观察验证数据的错误率，和最终模型的效果。



第十周 Up and Running with TensorFlow

- 课堂笔记

TensorFlow程序通常分为两部分：第一部分构建计算图谱（构造阶段），第二部分运行图谱（执行阶段）。 建设阶段通常构建一个表示ML模型的计算图谱，然后对其进行训练、计算。执行阶段通常运行循环，重复地评估训练步骤（例如每个mini-batch），逐渐改进模型参数。
管理图谱：创建的任何节点都会自动添加到默认图形中。若需要管理多个独立图形，可以创建一个新的图形并暂时将其设置为一个块中的默认图形（tf.reset_default_graph）。
Linear Regression with TensorFlow：TensorFlow操作可以采用任意数量的输入并产生任意数量的输出。Tensors具有类型和形状。以之前加州房屋数据集为例，使用进行TensorFlow线性回归。创建两个TensorFlow常数节点X和y来保存该数据和目标，并使用TensorFlow提供的矩阵运算来定义并评估theta。
Mini-batch Gradient Descent：每次迭代时用下一个Mini-batch替换X和Y，使用placeholder，它实际上并不执行任何计算，只是输出在运行时输出的数据，可以在训练时将训练数据传递TensorFlow。
储存、恢复模型：在创建完所有变量节点之后，创建一个Saver，在执行阶段，调用save() 方法。恢复模型时，在执行阶段的开始，调用restore() 方法。
TensorBoard：可以展示图形和训练曲线。在日志目录名称中包含时间戳，使得每次运行程序时使用不同的日志目录。

- 上机练习

安装TensorFlow，运行书上第九章代码，使用TensorBoard查看训练曲线。

- 课后工作

使用TensorFlow实现MNIST数据集上进行手写体识别分类（第六周上机）。
在TensorFlow中实现了softmax回归，直接调用softmax函数即可。softmax函数的本质是将一个K维的任意实数向量压缩（映射）成另一个K维的实数向量，其中向量中的每个元素取值都介于（0，1）之间，然后可以根据元素取值的大小来进行多分类的任务，如取权重最大的一维。 仅使用softmax函数，没有用到mnist.validation中的数据，实现MNIST识别，通过使用InteractiveSession类，更加灵活地构建代码。由于softmax回归模型是简单的模型，所以在MNIST数据集上正确率仅为0.9183。

第十一周 Introduction to Artificial Neural Network

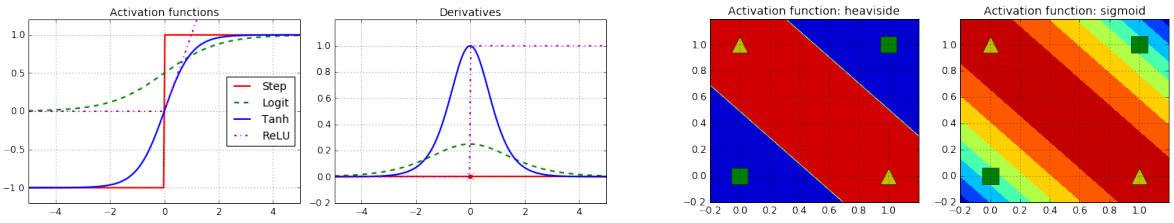
- 课堂笔记

第十章介绍了人工神经网络。神经网络起源于尝试让机器模仿大脑的算法，在80年代和90年代早期非常流行，慢慢在90年代后期衰落，最近由于计算机硬件能力提升，又开始流行起来。
人工神经元：有一个或更多的二态输入，实现一个二态输出。神经网络的目标就是模拟大脑中的神经元或网络。
感知机：一个感知机需要几个输入，x1，x2，…，然后产生单一的二进制输出，使用实数w1，w2来表示各个输入对输出的重要程度，神经元输出0还是1由输入的加权值是否超过阈值决定。通过调整权重和阈值（偏移），能够实现不同的决策模型。
多层感知机（MLP）：将输入的多个数据集映射到单一的输出的数据集上。多层感知器有至少一个隐藏层，单层感知器只能学习线性函数，而多层感知器也可以学习非线性函数。
反向传播（BP）算法：相邻层节点的连接都有权重。学习的目的是为这些边缘分配正确的权重。通过输入向量，这些权重可以决定输出向量。最初，所有的边权重都是随机分配的。对于所有训练数据集中的输入，人工神经网络都被激活，并且观察其输出。将输出和已知的、期望的输出进行比较，误差会传播回上一层。误差被标注，权重也会被相应的调整。重复该过程，直到输出误差低于阈值。
激活函数：ReLU (z) = max (0, z)；tanh (z) = 2σ(2z) – 1 ; logistic: $\sigma(z) = 1 / (1 + \exp(-z))$

- 上机练习

运行提供的第十章代码。

① 使用鸢尾花iris数据集来训练感知机模型，参数选择petal length和petal width。定义不同激活函数并绘制图像。使用多层感知机实现异或（XOR）逻辑，并绘制激活函数使用阶跃函数以及sigmoid函数的模型等高线结果。



② 使用TensorFlow中提供的DNNClassifier训练MNIST数据集，获得结果accuracy约为0.9825。

③ 不使用包装好的DNN分类器，而使用TensorFlow自己编写DNN模型用于训练MNIST数据集。自己定义的DNN模型包含两个隐藏层，激活函数都是ReLu。根据稀疏表示的label和输出层数据计算损失，并且使用梯度下降进行优化，得到结果accuracy约为0.98。

第十二周 Training Deep Neural Nets

- 课堂笔记

梯度消失、梯度爆炸：反向传播算法的工作原理是从输出层到输入层，传播梯度。 一旦该算法已经计算了网络中每个参数的损失函数的梯度，它就使用这些梯度来用梯度下降步骤来更新每个参数。梯度消失是指梯度下降更新使得低层连接权重实际上保持不变，并且训练永远不会收敛到良好的解决方案。 在某些情况下，可能会发生相反的情况：梯度可能变得越来越大，许多分层得到疯狂的权重更新，算法发散，这是梯度爆炸的问题。

批量标准化：在每层的激活函数之前在模型中简单地对输入进行零中心和归一化，然后使用每个层的两个新参数对结果进行缩放和移位，让模型学习到每层输入值的最佳尺度和均值。

梯度剪裁：减少爆炸梯度问题的一种方法是在反向传播过程中简单地剪切梯度，使它们不超过某个阈值。

无监督预训练：如果有很多未标记的训练数据，可以逐层训练图层，从最低层开始，然后上升，使用无监督的特征检测器算法， 每个图层都被训练成先前训练过的图层的输出，一旦所有图层都以这种方式进行了训练，就可以使用监督式学习对网络进行微调。

提前停止：为避免过度拟合训练集，一个很好的解决方案就是尽早停止训练（在第4章中也有提到）：只要在训练集的性能开始下降时停止训练。

Drop out：在每个训练步骤中，每个神经元（包括输入神经元，但不包括输出神经元）都有一个暂时“退出”的概率p，这意味着在这个训练步骤中它将被完全忽略， 在下一步可能会激活。超参数p称为丢失率，通常设为50%。 训练后，神经元不会再下降。

- 上机练习

运行第十一章代码。

① 绘制不同的激活函数sigmoid、Leaky ReLU、ELU、SELU的图像，并了解不同激活函数处理消失/爆炸梯度问题的能力以及训练时的优缺点。

② 针对MNIST数据集建立神经网络，每层都使用ELU激活函数，并且使用批量标准化的方法进行训练，再经过20轮训练后，在测试集上得到0.9652的正确率，这对MNIST来说不是一个很好的准确性。 如果训练的时间越长，准确性会变好，但是由于这是一个非常浅的网络，批量标准化和ELU不太可能产生非常积极的影响。

第十三周 Convolutional Neural Networks

- 课堂笔记

卷积层：第一层卷积层中的神经元不连接到输入图像中的每一个像素，而是连接到它们的局部感受野中的像素；而第二层卷积层中的每个神经元只与位于第一层中的小矩形内的神经元连接。 这种架构使得网络专注于第一隐藏层中的低级特征，然后将其组装成下一隐藏层中的高级特征。通过将局部感受野隔开，可以将较大的输入层连接到更小的层。
Filters：神经元的权重可以表示为局部感受野大小的小图像。使用相同filter的一个充满神经元的图层将提供一个特征图，该特征图突出显示图像中与filter最相似的区域。 在训练过程中，CNN试图找到最有用的过滤器，并学习将它们组合成更复杂的模式。
叠加特征图：在一个特征映射中，所有神经元共享相同的权重和偏置，不同的特征映射可能具有不同的参数。 卷积层同时对其输入应用多个滤波器，使其能够检测输入中的任何位置的多个特征。计算卷积层中给定神经元的输出是计算所有投入的加权总和加上偏置。
内存需求：CNN的卷积层需要大量的RAM，特别是在训练期间，因为反向传播的反向传递需要在正向传递期间计算的所有中间值。
池化层：目标是对输入图像进行二次抽样来减少计算负担、内存使用和参数数量，从而限制过拟合的风险。 就像在卷积层中一样，池化层中的每个神经元都连接到前一层中有限数量的神经元的输出，位于一个小的矩形感受域内，也必须像以前一样定义其大小，跨度和填充类型。

- 上机练习

运行提供的第十三章代码。

① 使用sklearn的load_sample_images加载两个样本图像china.jpg、flower.jpg，然后创建两个7×7的卷积核，并将它们应用到两张图形中，使用TensorFlow的conv2d函数构建的卷积图层，使用零填充且步幅为2，最后，绘制结果特征图，观察垂直线和水平线两种卷积核的作用。
② 观察池化层的作用，使用2×2内核创建最大池化层，跨度为2，没有填充，然后将其应用于之前加载的样本图像china.jpg，绘制结果图。注意这样的池化层输出在两个方向上都会减小两倍，所以它的面积将减少四倍，减少了75%的输入值。

- 课后工作

① 由于TensorFlow是一个非常强大的用来做大规模数值计算的库，因此TensorFlow适合于实现以及训练神经网络。使用TensorFlow为MNIST构建一个卷积神经网络，测试正确率，与第十周的softmax回归模型对比。使用ReLU神经元，用一个较小的正数来初始化偏置项，以避免神经元节点输出恒为0的问题。卷积的参数为：stride size=1，no padding，保证输出和输入是同一个大小。pooling操作采取max pooling，2*2 pooling kernel。第一层卷积由一个卷积接一个max pooling完成。卷积在每个5x5的patch中算出32个特征。把图片向量和权值向量进行卷积，加上偏置项，然后应用ReLU激活函数，最后进行max pooling。把几个类似的层堆叠起来，在第二层中，每个5x5的patch会得到64个特征。加入一个有1024个神经元的全连接层，用于处理整个图片。为了减少过拟合，在输出层之前加入dropout。最后，添加一个softmax层，和之前仅使用softmax回归模型一致。进行训练和评估时，使用ADAM优化器来做梯度最速下降，最后得到在测试集上的准确率约为99.2%，结果比第十周softmax回归模型、第六周的各种分类器以及第十一周的DNN模型的效果都好。

② 小组继续完善课程大作业Aicar的功能，实现简单物体识别，可以根据语音指令，寻找周围的目标物体并前往目标物体位置。目标检测的模型为Single Shot MultiBox Detector（SSD），使用单个深层神经网络检测图像中对象，将边界框的输出空间离散化为一组默认框，该默认框在每个特征图位置有不同的宽高比和尺寸。在预测期间，网络针对每个默认框中的每个存在对象类别生成分数，并且对框进行调整以更好地匹配对象形状。另外，网络组合来自具有不同分辨率的多个特征图的预测，以适应处理各种尺寸的对象。根据SSD模型识别出的特征图位置，给小车发送不同数字指令，控制小车前进的方向。