北京大学量化交易协会2019级培训

机器学习

沈廷威 胡天锐 胡磊 邓珂雅

2019-11-30

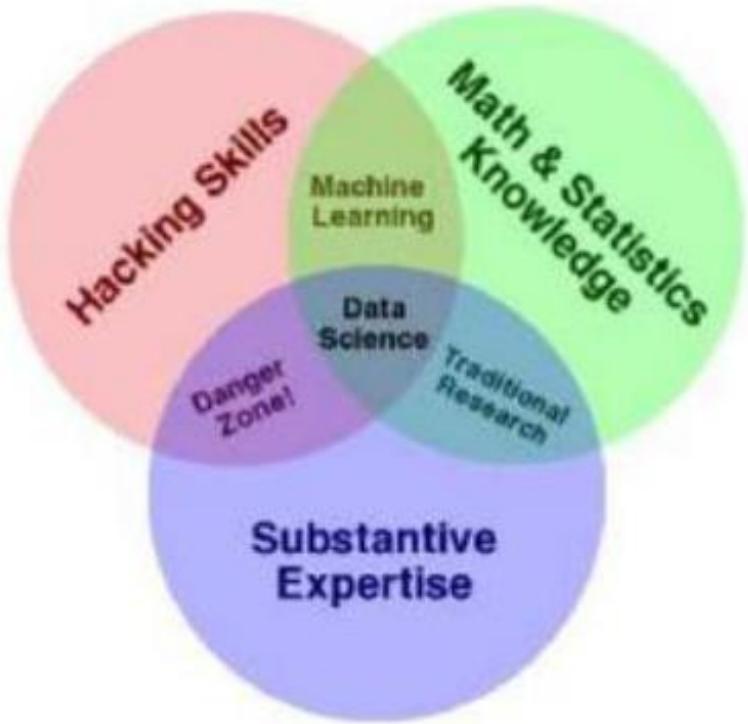
机器学习

机器学习

机器学习概述

概念

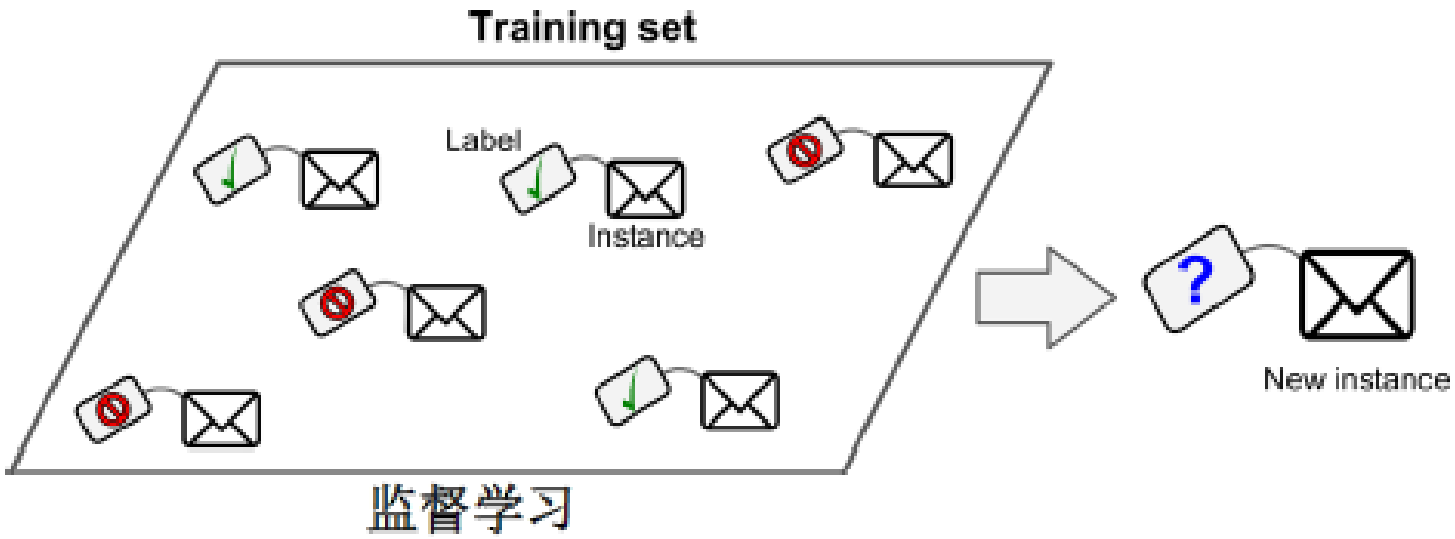
- 交叉学科，涉及概率论、统计学、逼近论、算法复杂度理论。
- 模拟或实现人类的学习行为，以获取新的知识或技能。



常见的机器学习问题

问题分类

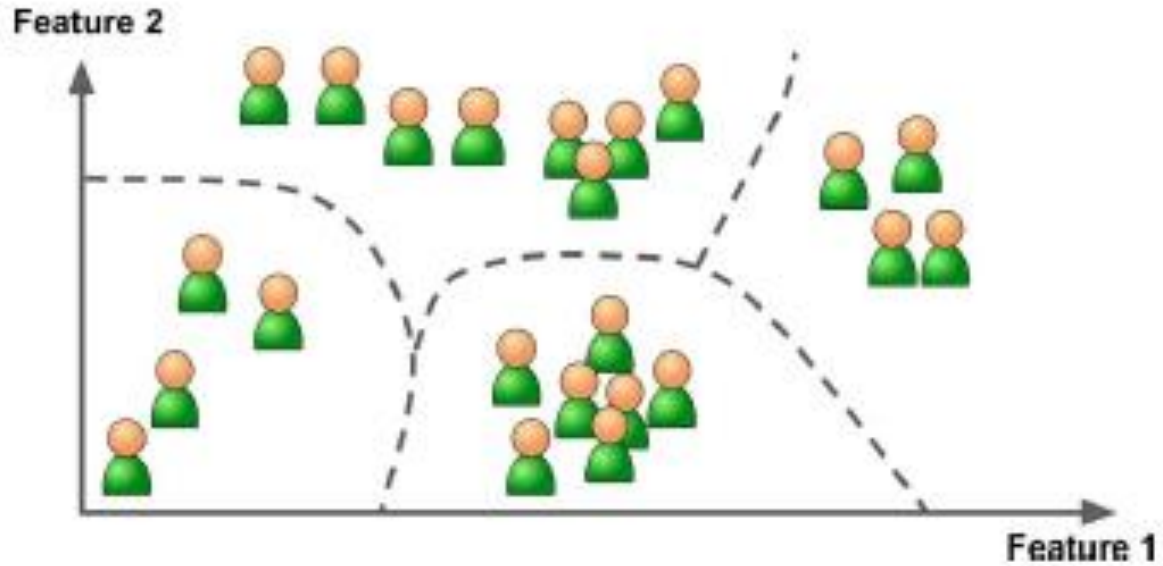
- 监督学习: 已知目标变量的分类信息。算法提供历史数据（输入和输出变量），并试图找到对样本外数据具有最佳预测能力的关系
- 应用: 手写文字识别，垃圾邮件分类



常见的机器学习问题

问题分类

- 无监督学习:样本无类别信息，需要学习器在训练数据中寻找规律
- 应用：社交网站分析

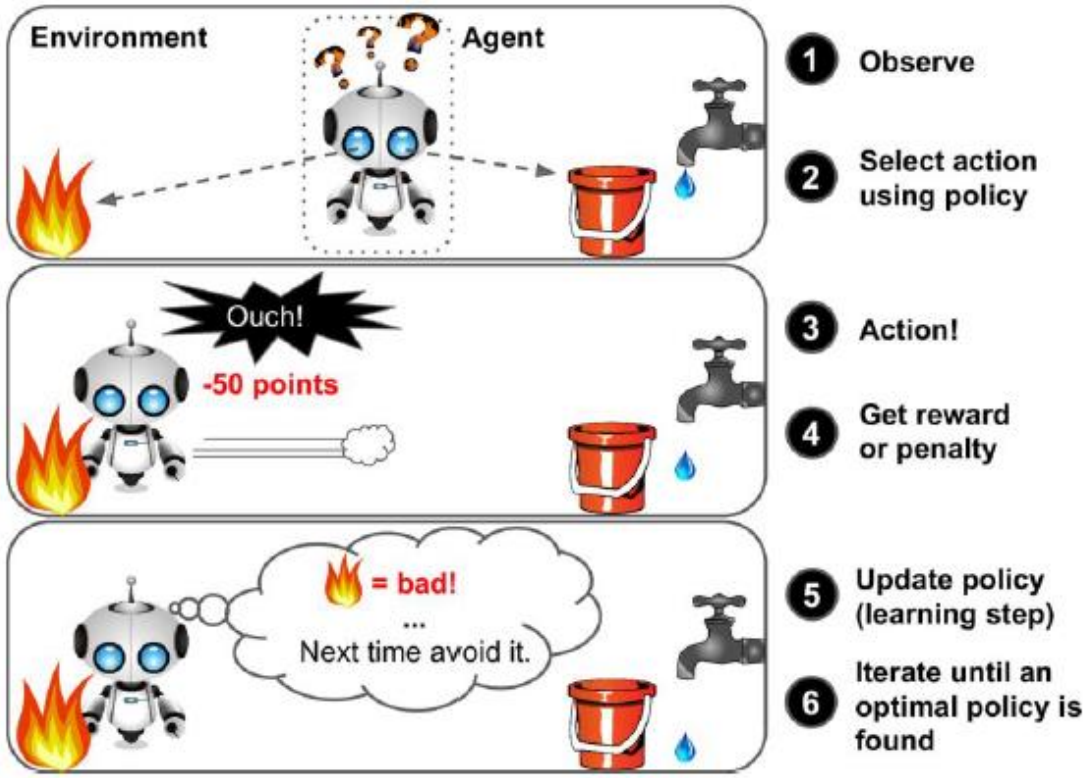


非监督学习

常见的机器学习问题

问题分类

- 强化学习: 在这种情况下称为代理(agent)的学习系统可以观察环境，选择和执行操作，并获得回报，以负面奖励的形式处罚。
- 应用 : 工业自动化

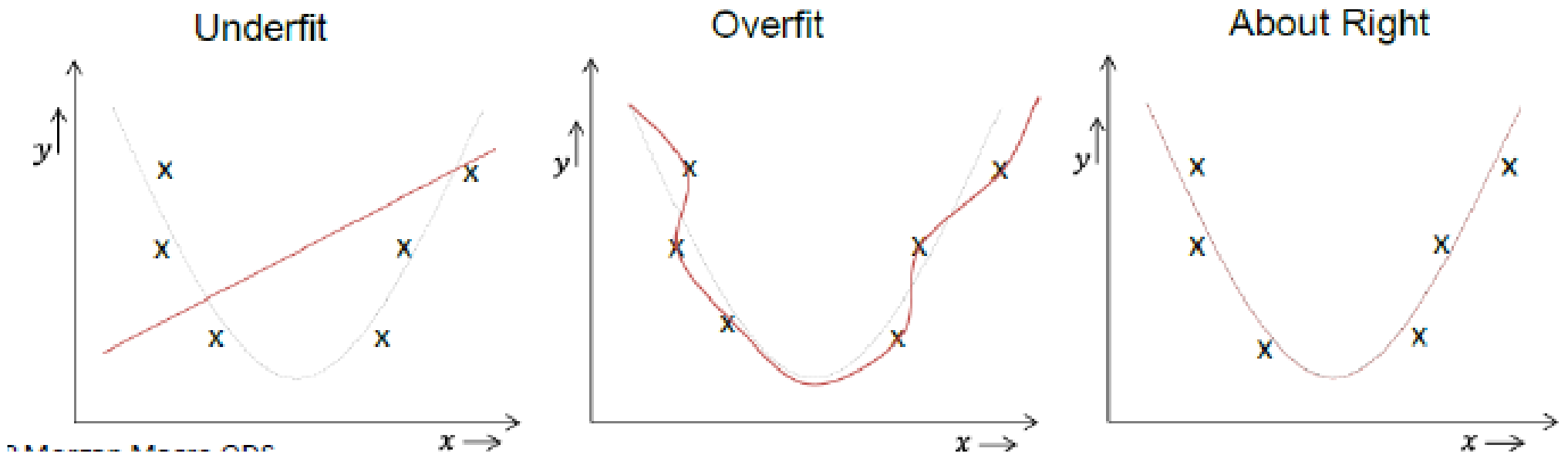


强化学习

模型的选择

欠拟合与过拟合

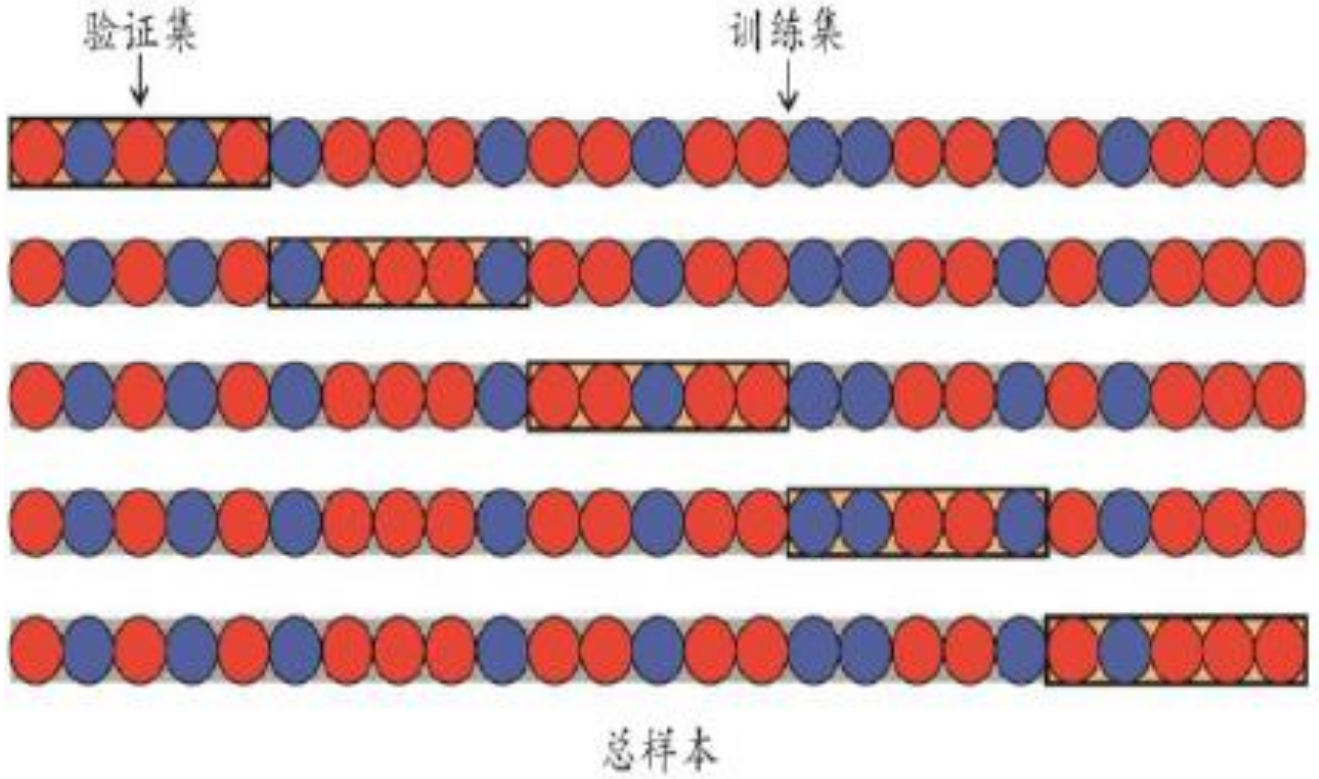
- 欠拟合：模型过于简单，对训练数据缺乏解释能力
- 过拟合：模型较为复杂，对训练数据解释性较好，对样本外数据缺乏解释力



数据集的划分

三种数据集

- 训练集:用于模型拟合的数据样本
- 验证集:是模型训练过程中单独留出的样本集，它可以用于调整模型的超参数和用于对模型的能力进行初步评估
- 测试集:用来评估模最终模型的泛化能力。
- 划分
  - 留出法
  - 交叉验证
  - bootstrapping



数据集的划分

三种数据集

- 训练集:用于模型拟合的数据样本
- 验证集:是模型训练过程中单独留出的样本集，它可以用于调整模型的超参数和用于对模型的能力进行初步评估
- 测试集:用来评估模最终模型的泛化能力。
- 划分
  - 留出法
  - 交叉验证
  - bootstrapping

减少因训练样本规模不同而导致的估计偏差

$$\lim_{m\mapsto\infty}\left(1-\frac{1}{m}\right)^{m}\mapsto\frac{1}{e}\approx 0.368$$

机器学习

1

2 决策树

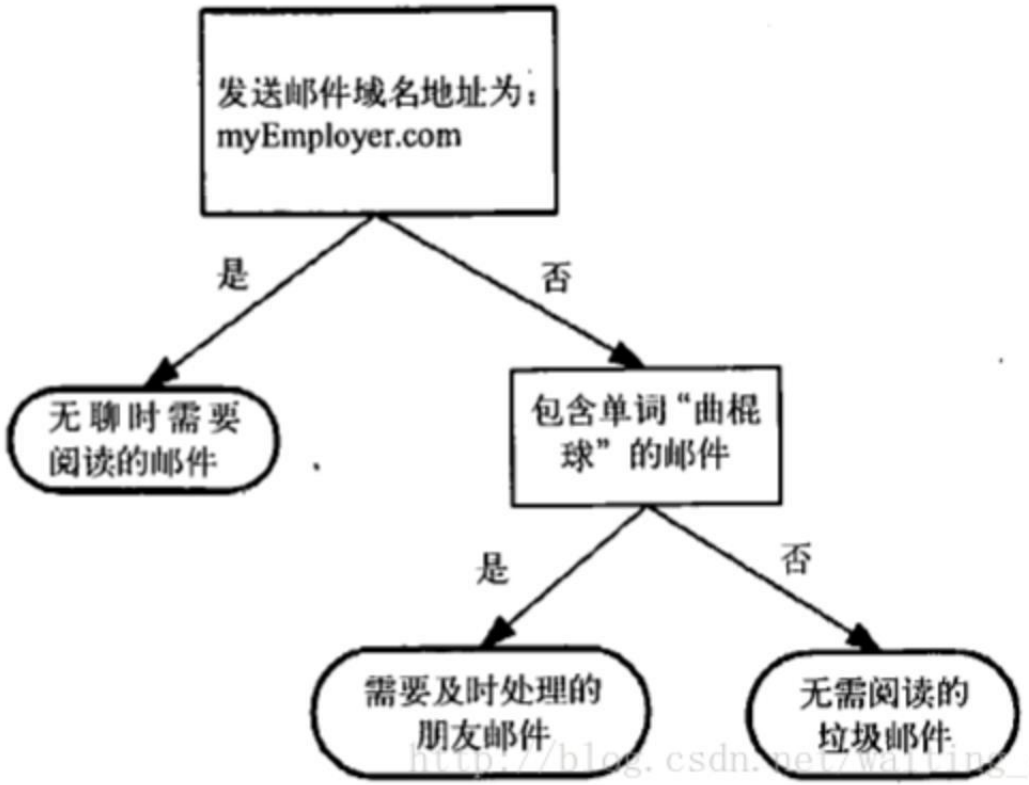
3

4

决策树

主要思想

- 从数据集合中，提取出一系列的规则，基于特征对实例进行分类的过程，可以认为是if-then的集合
- 核心思想就是找出更加纯净的子集，最好每个子集里都是结论极其单一的数据。



决策树

纯度判别方法

- ID3: 信息增益（最大）
- C4.5: 信息增益率（最大）
- CART: Gini指数（最小）

信息熵

- 又叫香农熵，是香农在1948年引入的一个概念，他指出，一个系统越是有序，信息熵就越低，一个系统越混乱信息熵就越高，信息熵被认为是一个系统有序程度的度量。
- 假定目标集合S中有n种样本，第k种样本所占比例为pk（k=1,2,3,...,n）,则S的信息熵为：

$$\mathrm{Ent}(S) = -\sum_{k=1}^{n} p_k \log_2 p_k$$

- 信息增益:Gain(k)=分裂前目标变量的信息熵-对特征值k分裂后的目标变量信息熵。若总数据量为D，分裂后数据成为有M个叶节点的分叉树，那么Gain（k）为:

$$\mathrm{Gain}(k) = \mathrm{Ent}(S) - \sum_{m=1}^{M} \mathrm{Ent}(D_m)$$

决策树

纯度判别方法

- ID3: 信息增益（最大）
- C4.5: 信息增益率（最大）
- CART: Gini指数（最小）

基尼指数

- 基尼值(S)指的是从数据S中随机取两个值属于不同类别的概率，因此Gini值越大意味着数据越混乱。

$$\mathrm{Gini}(S) = 1 - \sum_{k=1}^{n} p_k^2$$

- 若总数据量为D，分裂后数据成为有M个叶节点的分叉树，那么Gini_index(k)基尼指数为:

$$\text{Gini-index}(k) = \sum_{m=1}^{M} \mathrm{Gini}(D_m)$$

决策树

算法流程

- 1，寻找最适合分割的特征，最优的分割点，基于这一特征把数据分割成纯度更高的数据分组

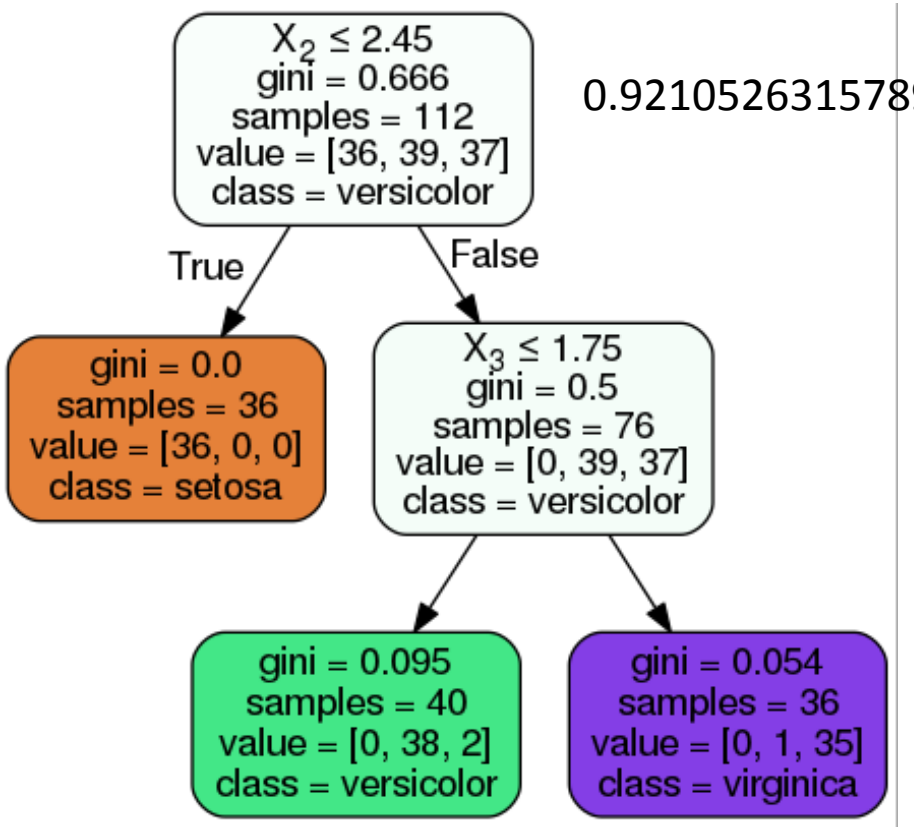
- 2，判断是否达到要求，若未达到，重复步骤一继续分割，直到达到要求停止为止。
  - 当子节点中只有一种类型的时候停止构建（此时该节点纯度最高，但很容易过拟合）
  - 当前节点样本数小于某个值，或者决策树深度次数达到指定值，停止构建，此时使用该节点中出现最多的类别样本数据作为对应值（常用）

- 3，剪枝，防止过拟合。
  - 预剪枝:在生成决策树过程中，对节点划分先进性预估，若节点分叉不能使决策树得到泛化提升，则停止分叉生成叶。
  - 后剪枝 :在生成一颗完整的决策树后，自下而上对每个叶片和节点进行评估，若减少分叉可以使决策树得到泛化提升，则转化分叉为一个叶。

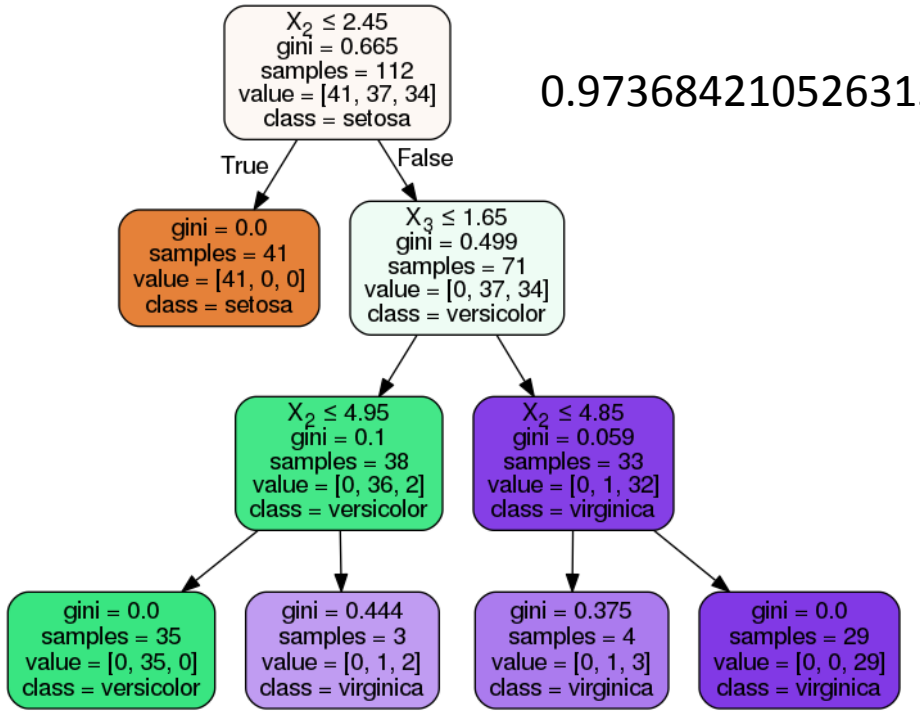
决策树

演示

- 使用sklearn提供的iris鸢尾花数据集,
- 四个特征：sepal length, sepal width, petal length, petal width
- 三个类别：Iris-Setosa, Iris-Versicolour, Iris-Virginica



0.9210526315789473



0.9736842105263158

随机森林

主要思想

- 随机森林就是通过集成学习的思想将多棵树集成的一种算法，它的基本单元是决策树，而它的本质属于机器学习的一大分支——集成学习（Ensemble Learning）方法。
- 每棵决策树都是一个分类器（假设现在针对的是分类问题），那么对于一个输入样本，N棵树会有N个分类结果。而随机森林集成了所有的分类投票结果，将投票次数最多的类别指定为最终的输出（若干个弱分类器组成一个强分类器）
- 随机的体现:
  - Bootstrapping方法生成训练集
  - 如果每个样本的特征维度为M，指定一个常数m<<M，随机地从M个特征中选取m个特征子集，每次树进行分裂时，从这m个特征中选择最优的。（影响树的相关性和分类能力）

随机森林

演示

- 使用2010年6月至2016年6月，上证指数收盘价的30日简单移动平均SMA，30日加权移动平均WMA，10日动量MOM作为特征
- 若第二日股价上涨，则归类为False，否则归类为True
- 使用规模为50的随机森林与单棵决策树分别进行训练，随机划出25%的数据作为测试集。

Forest:0.510989010989011

Tree:0.476648351648351 7

机器学习

1

2

3 **支持向量机**

4

机器学习：支持向量机SVM

1 支持向量机方法简介

2 线性支持向量机

3 核支持向量机

4 SVM在A股数据分析上的应用

支持向量机方法简介

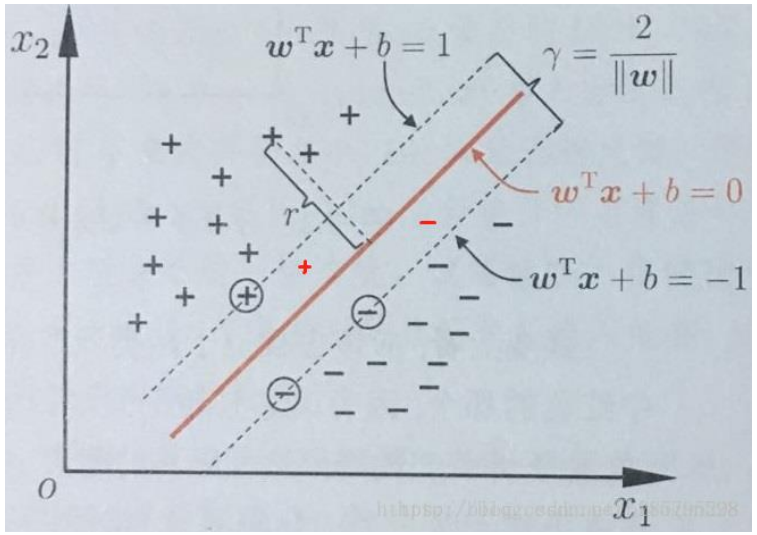
概念

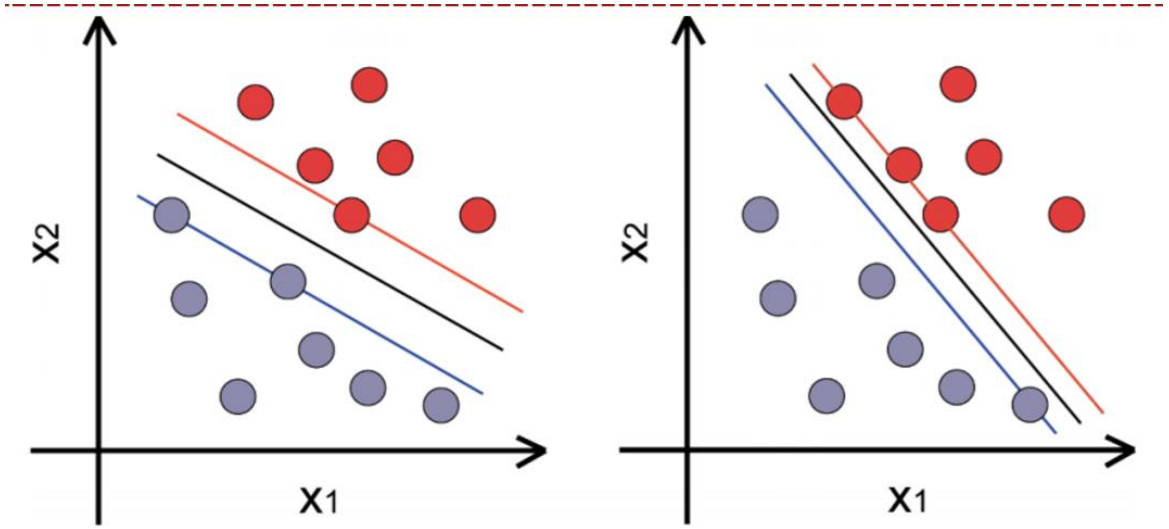
- 支持向量机(Support Vector Machine，SVM)，是一种二分类模型。属于监督学习范畴。
- 其目的是寻找 一个超平面来对样本进行分割，分割的原则是支持向量与超平面之间的距离最大化。
- 支持向量机可分为线性支持向量机和核支持向量机，前者针对线性分类问题，后者属于非线性分类器。

线性支持向量机

需处理的问题

- 给定训练样本集 $D=(x_1,y_1),(x_2,y_2),...,(x_m,y_m))),y_i\in\{-1,1\}$，线性分类器基于训练样本D 在二维空间中找到一个超平面来分开二类样本。
- ＋和－是要区分的两个类别，在二维平面中它们的样本如图所示。中间的直线就是一个分类函数，它可以将两类样本完全分开。一般的，如果一个线性函数能够将样本完全正确的分开，就称这些数据是线性可分的，否则称为非线性可分的。
- 超平面不唯一，“最优”的评价标准是最大化支持向量到该超平面的距离





线性支持向量机

问题的数学转化

- 超平面的方程为$w^T x + b = 0$
- 设分类超平面为 $w^T x + b = 0$ 两个最大边缘超平面方程分别为 $w^T x + b = 1$，-1
- 两个最大边缘超平面之间的距离：$\frac{2}{\|w\|}$

- 因此得到的等价的优化问题：

$$\max_{w,b} \frac{2}{\|w\|},$$

$$s.t.$$

$$y_i(w^T x_i + b) \geqslant 1, i = 1, ..., m$$

线性支持向量机

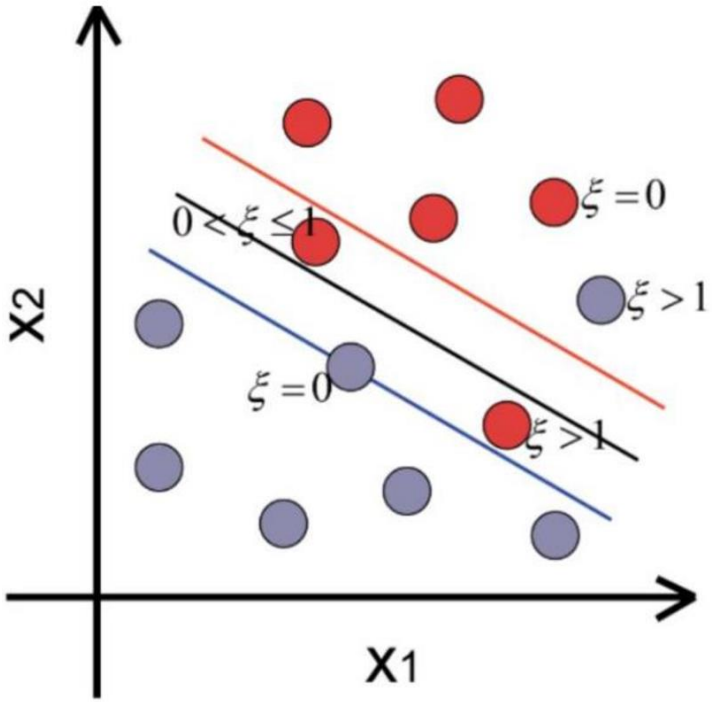
更实际的问题

- 现实中由于噪音等扰动，问题往往不是线性可分的
- 因此需要引入松弛变量，表示允许一定的错误率。
- 对于每个样本点赋予一个松弛变量，优化问题如下所示：

$$\min_{w,b} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i$$
$$\text{s.t.} \quad y_i\left(w^\top X_i + b\right) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

- 数学求解：定义拉格朗日乘子 $\alpha = \{\alpha_1, \ldots, \alpha_N\}$, $\mu = \{\mu_1, \ldots, \mu_N\}$
  得到拉格朗日函数：

$$\mathcal{L}(w, b, \xi, \alpha, \mu) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i + \sum_{i=1}^{N}\alpha_i\left[1 - \xi_i - y_i\left(w^\top X_i + b\right)\right] - \sum_{i=1}^{N}\mu_i\xi_i$$



令 $\frac{\partial \mathcal{L}}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{N}\alpha_i y_i X_i$, $\frac{\partial \mathcal{L}}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N}\alpha_i y_i = 0$, $\frac{\partial \mathcal{L}}{\partial \xi} = 0 \Rightarrow C = \alpha_i + \mu_i$

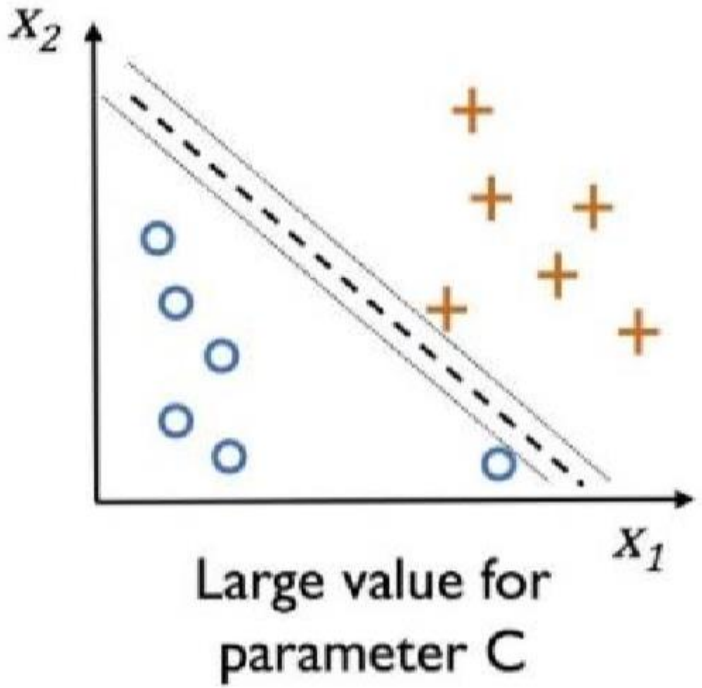
代回拉格朗日函数得到原问题的对偶问题

$$\max_{\alpha} \quad \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left[\alpha_i y_i (X_i)^\top \left(X_j\right) y_j \alpha_j\right]$$
$$\text{s.t.} \quad \sum_{i=1}^{N}\alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C$$

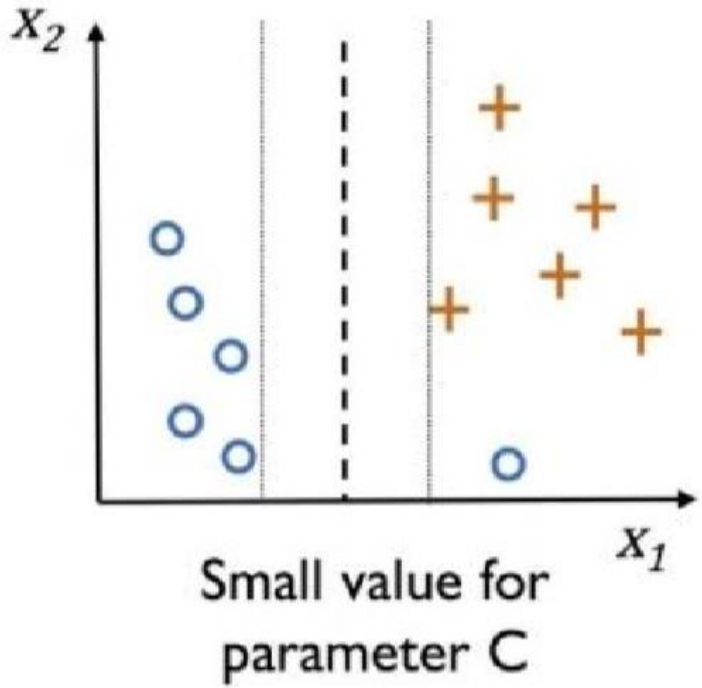
线性支持向量机

参数C的选择

- C的大小决定了得到的模型对于错误的容忍程度
- 较小的C取值意味着模型对于错误的容忍程度较大，反之则意味着对于错误的容忍较小
- C的选择需要在训练集上根据实际数据进行选择



Large value for parameter C

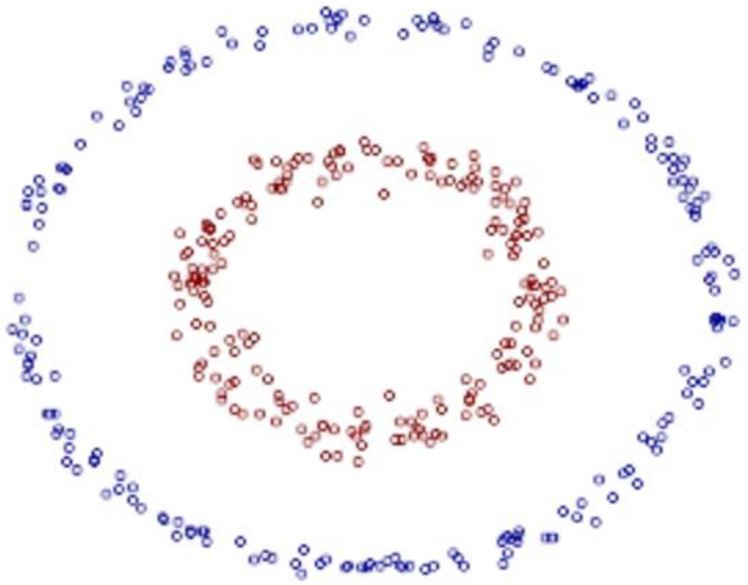


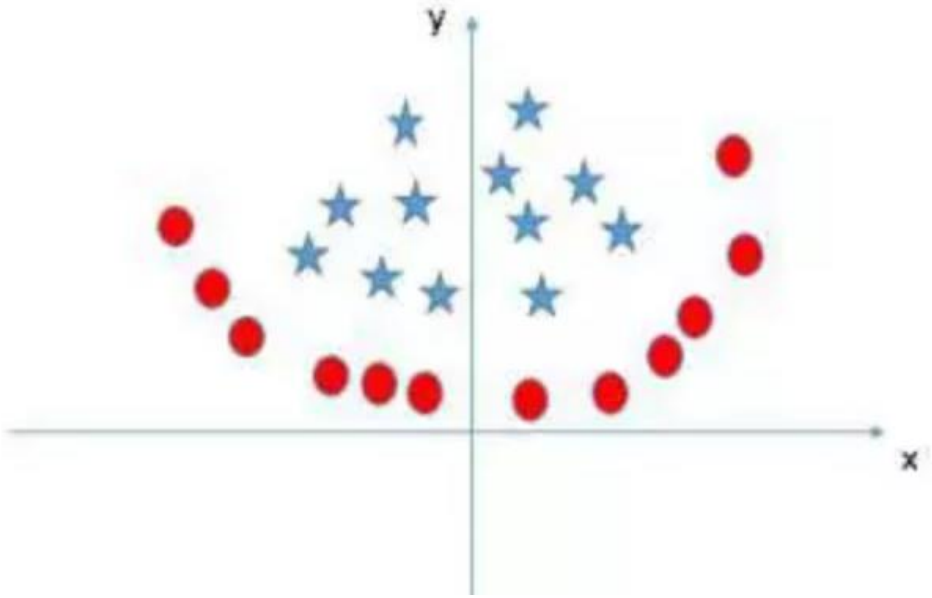
Small value for parameter C

核支持向量机

概念

- 考虑比上述需要引入松弛变量的问题更复杂的问题，如图所示
- 此时需要将数据映射到高维空间内 利用高维空间中的超平面来划分
- 考虑原始数据本身维度就较高的情况，可能会造成较高的计算时间
- 因此需要引入核函数简化计算





核支持向量机

核函数

- 考虑线性支持向量机时导出的原问题的对偶问题如下：

$$\max_{\alpha} \quad \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left[\alpha_i y_i (X_i)^{\top}\left(X_j\right) y_j \alpha_j\right]$$

$$\text{s.t.} \quad \sum_{i=1}^{N}\alpha_i y_i = 0, \quad 0 \le \alpha_i \le C$$

- 假设使用的映射为Φ，则相应的对偶问题如下：

$$\begin{aligned} min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(\phi_i \cdot \phi_j) - \sum_{i=1}^{N}\alpha_i \\ s.t. \quad & \sum_{i=1}^{N}\alpha_i y_i = 0 \\ \alpha_i \ge 0, \quad & i = 1,2,\ldots,N \end{aligned}$$

- 所谓的核函数就是 $\kappa(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$

核支持向量机

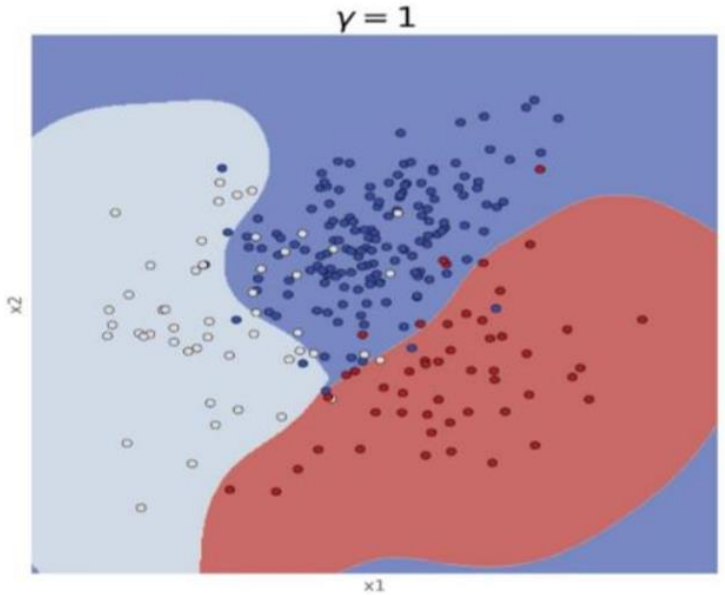
核函数

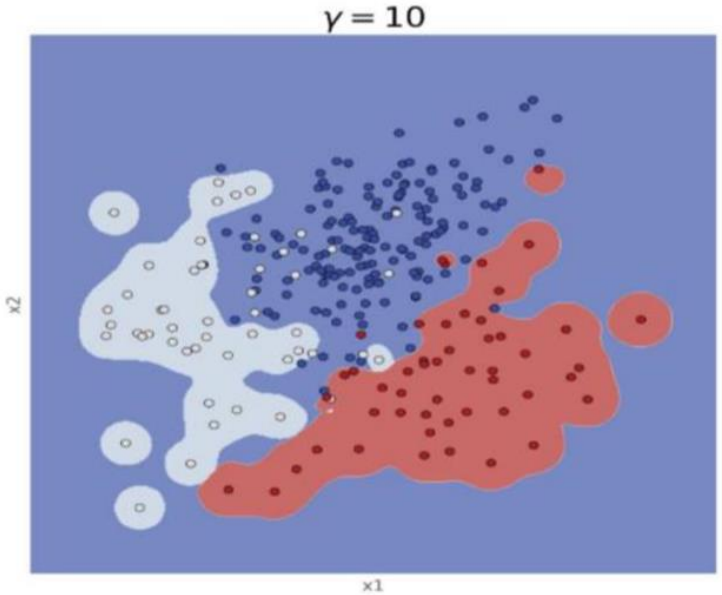
- 一些比较常用的核函数
- 线性核（Linear Kernel）$\kappa(x, x_i) = x \cdot x_i$
  主要用于线性可分的情况，在原始空间中寻找最优线性分类器，具有参数少速度快的优势。对于线性可分数据，其分类效果很理想，因此我们通常首先尝试用线性核函数来做分类，
- 多项式核（Polynomial Kernel）$\kappa(x, x_i) = ((x \cdot x_i) + 1)^d$
  适合于正交归一化（向量正交且模为1）数据。但是多项式核函数的参数多，当多项式的阶数d比较高的时候，由于学习复杂性也会过高，易出现过拟合现象。
- 径向基核函数（Radial Basis Function）$k(x, y) = \exp\left(-\gamma \|x - y\|^2\right)$
  也称为高斯核函数，径向基函数是一种局部性强的核函数，其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。
- Sigmoid核（Sigmoid Kernel）$\kappa(x, x_i) = tanh(\eta < x, x_i > + \theta)$
  采用Sigmoid核函数，支持向量机实现的就是一种多层神经网络。

核支持向量机

参数选择

- Sigmoid核和高斯核等核函数都包含γ，γ值决定了原始数据映射后，在高维特征空间中的分布。γ越大，样本在高维空间中的分布越稀疏，样本之间间隔越远，更容易被分类边界区分开来，因而训练集正确率更高，也更容易导致过拟合。
- 左图γ值较小的情况下，边界较为简单，模型基本能够区分三类样本，然而无法正确判别部分极端样本。右图γ值较大，分类边界极为复杂，分类器学习了更多极端样本，训练集正确率非常高，但是出现了过拟合。





核支持向量机

模型评价

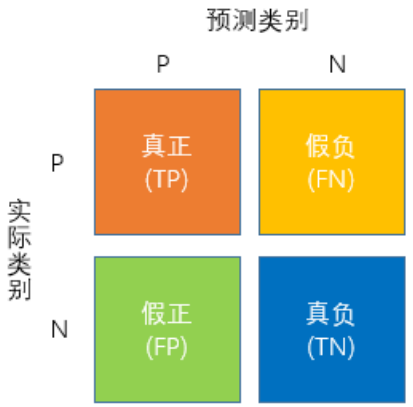
- 常使用混淆矩阵confusion matrix来展示分类学习算法的性能
- 参考指标：

准确率(Accuracy)： 衡量所有样本被分类准确的比例 $Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$

精确率(Precision)：预测为正样本的样本有多少是真的正样本 $Precision = \frac{TP}{TP + FP}$

召回率(Recall)： 表示分类正确的正样本占总的分类正确样本的比例 $Recall = \frac{TP}{TP + FN}$

F1-score： 精确率和召回率的调和平均 $\frac{2}{F_1} = \frac{1}{P} + \frac{1}{R} \Longrightarrow F_1 = \frac{2PR}{P + R} = \frac{2TP}{2TP + FP + TN}$



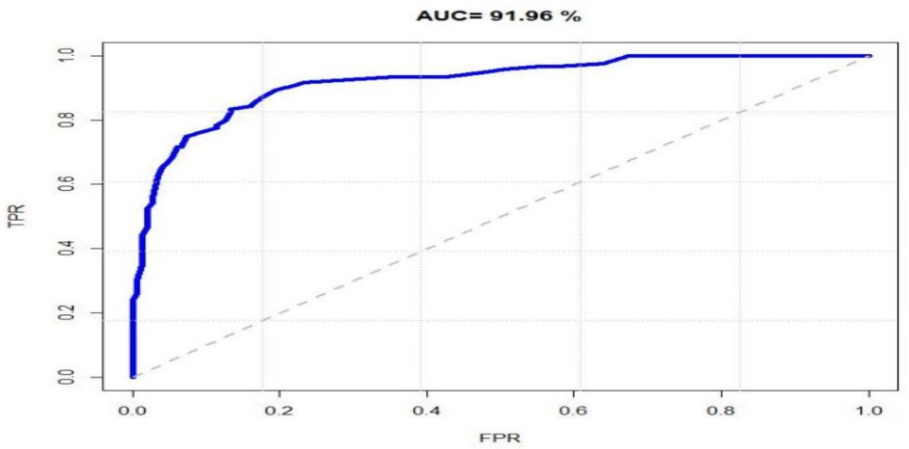
核支持向量机

模型评价

- ROC曲线也是常用的评价指标
- ROC曲线横坐标是FPR(False Positive Rate)，纵坐标是TPR(True Positive Rate)

$$FPR = \frac{FP}{FP + TN}, TPR = \frac{TP}{TP + FN}$$

- 一般来说，如果ROC是光滑的，那么基本可以判断没有太大的overfitting，AUC面积越大一般认为模型越好。AUC（Area Under Curve）被定义为ROC曲线下的面积，显然这个面积的数值不会大于1。又由于ROC曲线一般都处于y=x这条直线的上方，所以AUC的取值范围在0.5和1之间。使用AUC值作为评价标准是因为很多时候ROC曲线并不能清晰的说明哪个分类器的效果更好，而作为一个数值，对应AUC更大的分类器效果更好。



SVM方法在A股数据上的应用

数据选取与处理

- 收集的数据时间跨度为2014年2月25日至2019年4月22日，涉及的数据主要有沪深300指数每日高开低收和成交量。
- 随后对于原始数据计算相应的一些技术指标：实体、最高价差、上影线、下影线。随后最高价差作为分母，对实体，上影线，下影线作标准化修正。
- 使用Python中talib扩展包计算DIFF、DEA和hist指标

- 将原始数据集按照80：20的比例划分为训练集和测试集。按照每日指数的走向，如果T日收盘数据低于T-1日收盘数据，则令对应T日的标签Y取值为-1，反之则为1。
- 对于T日的标签Y，从T日向前打开一个长度为5的时间窗口，该时间窗口所涵盖的每一个交易日的Solid、Upshadow、Downshadow、DIFF、DEA、hist数据都作为分类变量。最终建立模型前还需要对其作标准化修正。

- 使用高斯核函数建模。对于惩罚系数C和高斯核参数gamma，我们对其进行调参，令C和gamma的的取值分别各自遍历0.001，0.01，0.1，1，10，100，得到最优参数为C=10,gamma=0.1。

- 在训练集上，模型的分类正确性达到100%，同时在测试集上，模型的分类正确性也达到了53.49%。

SVM方法在A股数据上的应用

相关代码实现

```
###划分训练集和测试集
Xtrain,Xtest,ytrain,ytest=train_test_split(X,y,test_size=0.2,random_state=0)


from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
Xtrainadj=scaler.fit_transform(Xtrain)
Xtestadj=scaler.fit(Xtrain).transform(Xtest)


###训练模型
clf=svm.SVC(kernel='rbf',C=10,gamma=0.1)
clf.fit(Xtrainadj,ytrain)

print(clf.score(Xtrainadj,ytrain))
print(clf.score(Xtestadj,ytest))
```

```
###调参
highscore=0
C=0
gamma=0

for m in[0.001,0.01,0.1,1,10,100]:
    for n in [0.001,0.01,0.1,1,10,100]:
        if check(m,n)>highscore:
            C=m
            gamma=n
            highscore=check(m,n)

print(C,gamma)
```

SVM方法在个股数据上的应用

数据选取与处理

- 收集的数据时间跨度为2017年1月3日至2019年1月18日。涉及的股票为A股市场中除去新三板等小盘股和数据接口未提供数据股票的所有个股，总计3136支个股。
- 预处理时将每十天作为一期，分别计算每一期的收益率，T期收益率记为P_T

- 选取的指标为市值、主力资金净流入两个数据，计算个股当期主力资金净流入，然后将其与第T期10个交易日内市值的均值作商。
- 将得到的商的绝对值取自然对数，并乘以sgn(个股第T期主力资金净流入/第T期10个交易日内市值的均值)

- SVM分类模型如下：对于个股j总计五十期的数据，将第T+1期的收益率与第T期的因子值对应起来，构建相应的支持向量X和标签Y，其中X即第T期因子值，第T+1期收益率若为正则Y为1，否则为-1。

- 在除去所有缺失的数据后，共余下141088组数据，将所有的X值进行标准化以使其近似服从于标准正态分布，然后按照80：20的比例将所有样本分类为训练集和测试集。

- 在80%的训练集上，使用线性核建立SVM模型，并在20%的测试集上进行测试，程序的结果输出显示分类的正确率可以达到54.83%，显著超过了平凡的预测正确率50%。该测试集的样本容量为28218。

机器学习

1

2

3

4 深度学习

机器学习：深度学习

1 机器学习方法论

2 前向传播与反向传播

3 卷积神经网络（CNN）

4 循环神经网络（RNN）

深度学习概述

机器学习方法论

- 在哲学上，经典机器学习方法在某种程度上可归属于还原主义。它通常是用人类的先验知识，把原始数据预处理成各种特征(feature)，然后对特征进行分类，这种分类的效果，高度取决于特征选取的好坏。所以传统机器学习又可以被称为特征工程。
- 我们逐渐发现可以让神经网络自己学习如何抓取数据的特征，这样的效果更好，这就是特征表示学习。但机器自己学习出来的特征存在于机器空间，这超越了人类理解的范畴，对我们是一个黑盒。
- 随着网络的进一步加深，出现多层次的“表示学习”，这就是Deep Learning。

提出卷积神经网络，改进反向传播算法

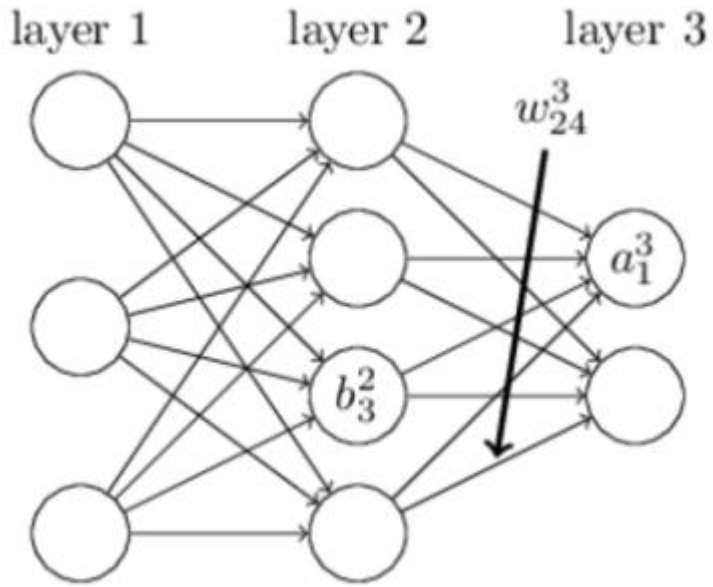
注意力机制，生成对抗网络

玻尔兹曼机，修正卷积神经网络

前向传播

神经网络的训练可以分为两个步骤：前向传播与反向传播

前向神经网络本质上是个多层感知机，其由输入层、输出层与若干隐含层构成，每个神经元模型都有自己的阈值与对应的激活函数。



左图是一个三层人工神经网络，layer1至layer3分别是输入层、隐藏层和输出层。

$\mathbf{W}_{jk}^{l}$表示第l-1层的第k个神经元连接到第l层的第j个神经元的权重；

$\mathbf{b}_{j}^{l}$表示第l层的第j个神经元的偏置；

$\mathbf{Z}_{j}^{l}$表示第l层的第i个神经元的输入，即：

$$z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l$$

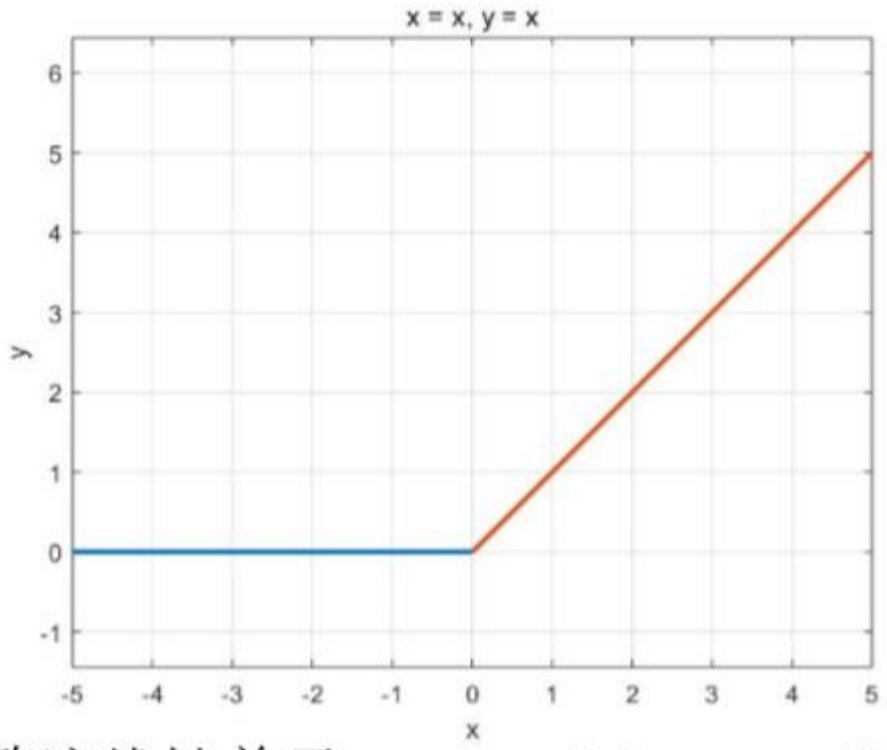
其中a_k^{l-1}表示第l-1层的第k个神经元的输出，即：

$$a_j^l = \sigma\left(\sum_k w_{jk}^l a_k^{l-1} + b_j^l\right)$$

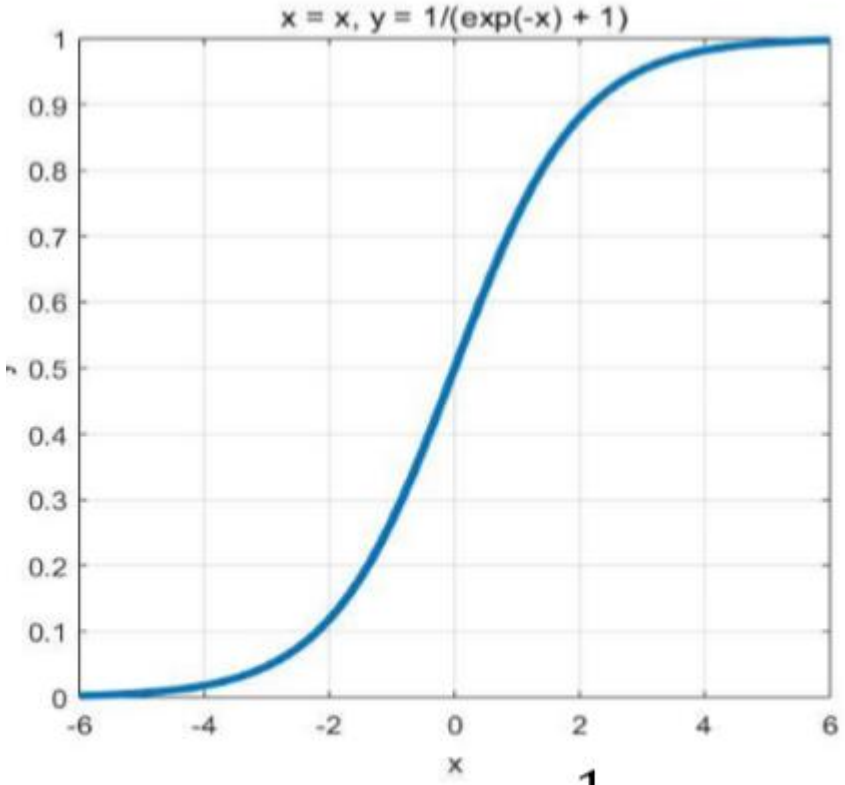
其中σ表示激活函数

前向传播网络

常用激活函数：阶跃函数、ReLU函数、sigmoid函数等。



整流线性单元ReLU：$g(z) = \max(0, z)$



$$sigmoid = \frac{1}{1 + e^{-x}}$$

前向传播网络

Cost function被用来计算ANN输出值与实际值之间的误差，常用的代价函数是二次代价函数

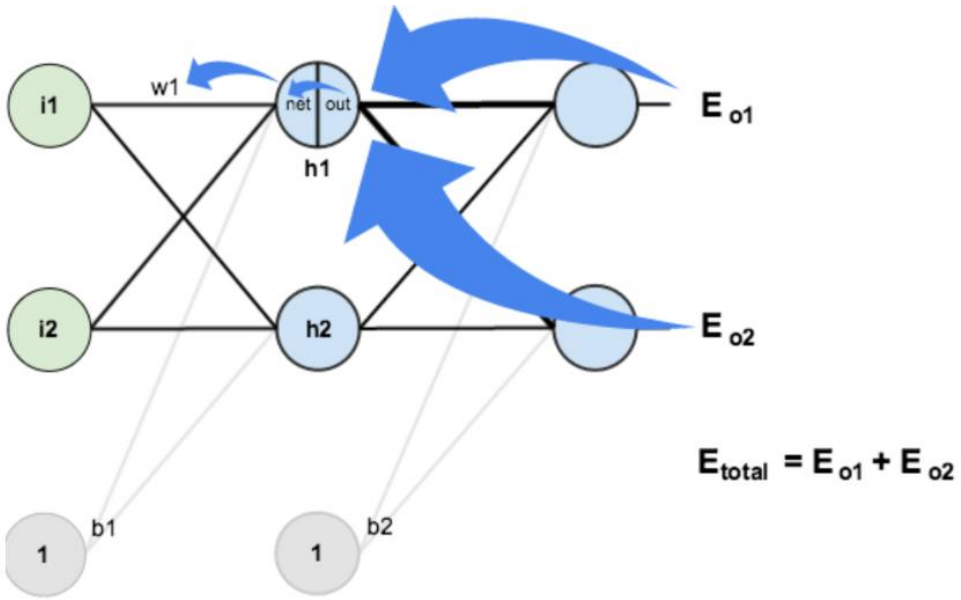
$$C = \frac{1}{2n}\sum_x \|y(x) - a^L(x)\|^2$$

其中，x表示输入的样本，y表示实际值，a^L 表示预测的输出，L表示神经网络的最大层数

将第l层第j个神经元中产生的错误(即实际值与预测值之间的误差)定义为：

$$\delta_j^l \equiv \frac{\partial C}{\partial z_j^l}$$

反向传播算法（BP）



Step1: 利用输出层计算出前向传播的错误（输出值与实际值的误差）

Step2: 设置learning rate，使用梯度下降、随机梯度下降等优化方法逐渐减小误差。

Step3: 利用链式法则进行反向传播，减小误差，不断修正网络各层的连接权值Wij

前向传播网络与反向传播

例子

西瓜数据集

编号	色泽	根蒂	敲声	纹理	脐部	触感	密度	含糖率	好瓜
1	2	2	2	1	3	1	0.697	0.46	1
2	3	2	3	1	3	1	0.744	0.376	1
3	3	2	2	1	3	1	0.634	0.264	1
4	2	2	3	1	3	1	0.608	0.318	1
5	1	2	2	1	3	1	0.556	0.215	1
6	2	1	2	1	2	2	0.403	0.237	1
7	3	1	2	2	2	2	0.481	0.149	1
8	3	1	2	1	2	1	0.437	0.211	1
9	3	1	3	2	2	1	0.666	0.091	0
10	2	3	1	1	1	2	0.243	0.267	0
11	1	3	1	3	1	1	0.245	0.057	0
12	1	2	2	3	1	2	0.343	0.099	0
13	2	1	2	2	3	1	0.639	0.161	0
14	1	1	3	2	3	1	0.657	0.198	0
15	3	1	2	1	2	2	0.36	0.37	0
16	1	2	2	3	1	1	0.593	0.042	0
17	2	2	3	2	2	1	0.719	0.103	0

前向传播网络与反向传播

```python
import pandas as pd
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
## 读取数
data = pd.read_excel(r'C:\Users\10295\Desktop\xigua3.xlsx')
## 将target变为数字

## 取出X和y
X = pd.get_dummies(data.iloc[:,1:-1]).values
y = data.iloc[:,-1].values
## 切割数据集
X_train,X_test,y_train,y_test = train_test_split(X,y,train_size=0.8,random_state=125)
## 建模并预测
BPNet = MLPClassifier(random_state=123)
BPNet.fit(X_train,y_train)
y_pred = BPNet.predict(X_test)
#print(y_test,y_pred)
# #输出预测结果报告
print('预测报告为：\n',classification_report(y_test,y_pred))
```

预测报告：

```
In [13]: runfile('E:/work/untitled0.py')
预测报告为：
              precision    recall  f1-score

           0       1.00      1.00      1.00
           1       1.00      1.00      1.00

avg / total       1.00      1.00      1.00
```

卷积神经网络(CNN)

基本概念

- 卷积神经网络（CNN）是一类包含卷积计算且具有深度结构的前馈神经网络是深度学习的代表算法之一。卷积神经网络具有表征学习（representation learning）能力，能够按其阶层结构对输入信息进行平移不变分类，因此也被称为“平移不变人工神经网络”。
- 卷积神经网络是指那些至少在网络的一层中使用卷积运算来替代一般的矩阵乘法的神经网络。
- CNN擅长处理具有类似网格结构的数据，例如图像数据（可以看做二维的像素网格）

- 数学概念：

连续型卷积：

$$s(t) = \int_{-\infty}^{\infty} x(a)w(t-a)da$$

离散型卷积：

$$s(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a)$$

x (a) 通常为模型的输入，w $(t-a)$称为核。

二维数据的卷积：

$$S(i,j) = \sum_{m}\sum_{n} I(m,n)K(i-n,j-n)$$

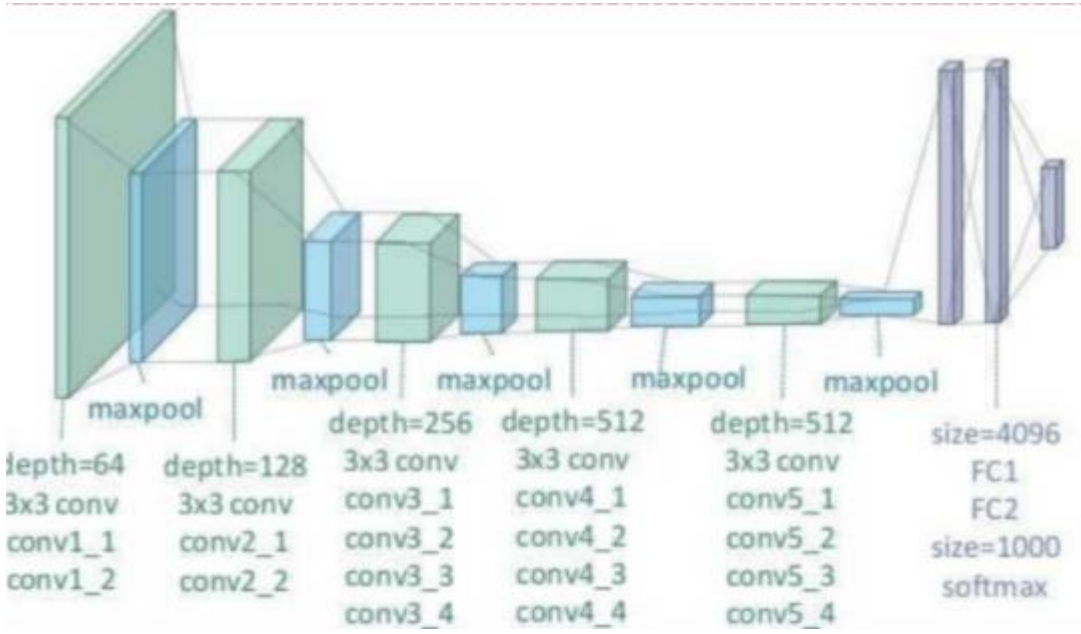
卷积神经网络(CNN)

池化（pooling）

• 为了有效地减少计算量，CNN使用的另一个有效的工具被称为“池化(Pooling)”。池化就是将输入图像进行缩小，减少像素信息，只保留重要信息。

池化的操作也很简单，通常情况下，池化区域是2*2大小，然后按一定规则转换成相应的值，例如取这个池化区域内的最大值（max-pooling）、平均值（mean-pooling）等，以这个值作为结果的像素值。

• 池化函数：使用某一位置的相邻输出的总体统计特征来作为该位置的输出。常用的池化函数包括最 大值、平均值、$L2$范数和基于据中心像素距离的加权平均数

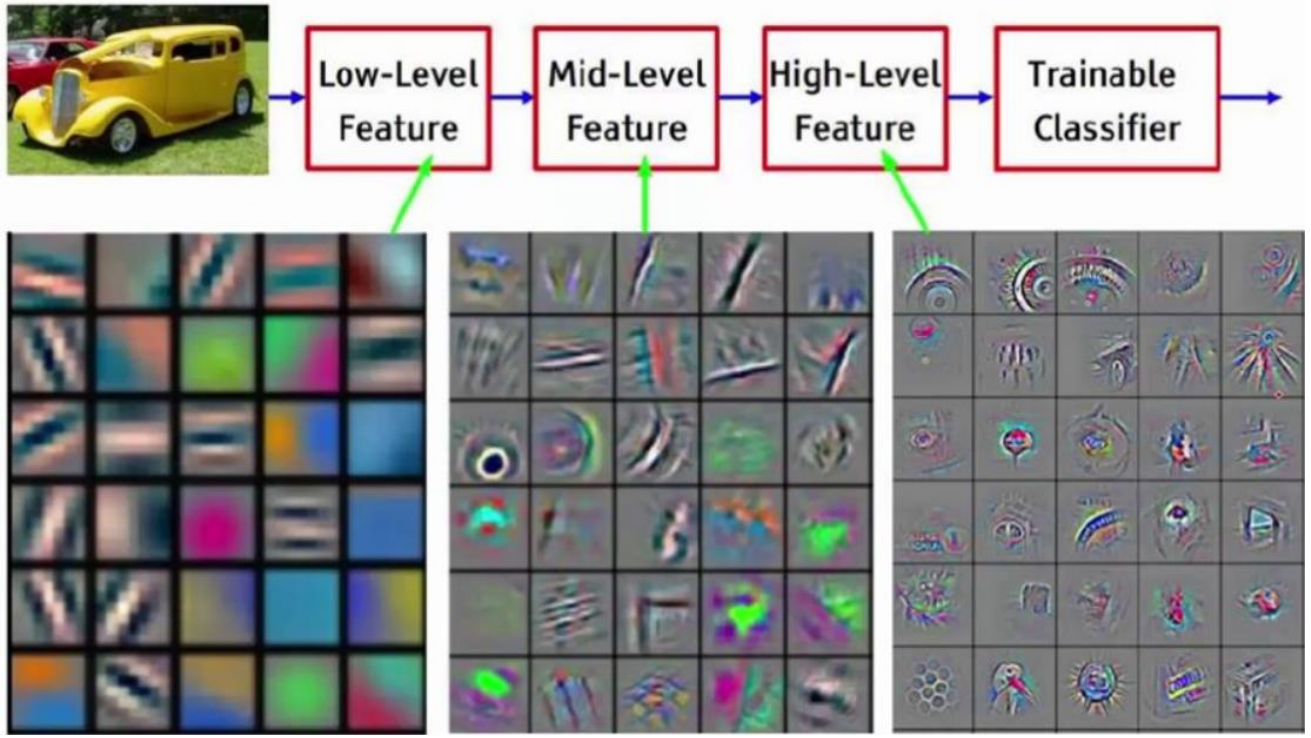


前向传播网络与反向传播

例子

卷积神经网络仿造生物的视知觉（visual perception）机制构建，可以进行监督学习和非监督学习，其隐含层内的卷积核参数共享和层间连接的稀疏性使得卷积神经网络能够以较小的计算量对格点化（grid-like topology）特征，例如对像素和音频进行学习。

直观理解卷积



以左图为例：

第一次卷积可以提取出低层次的特征。

第二次卷积可以提取出中层次的特征。

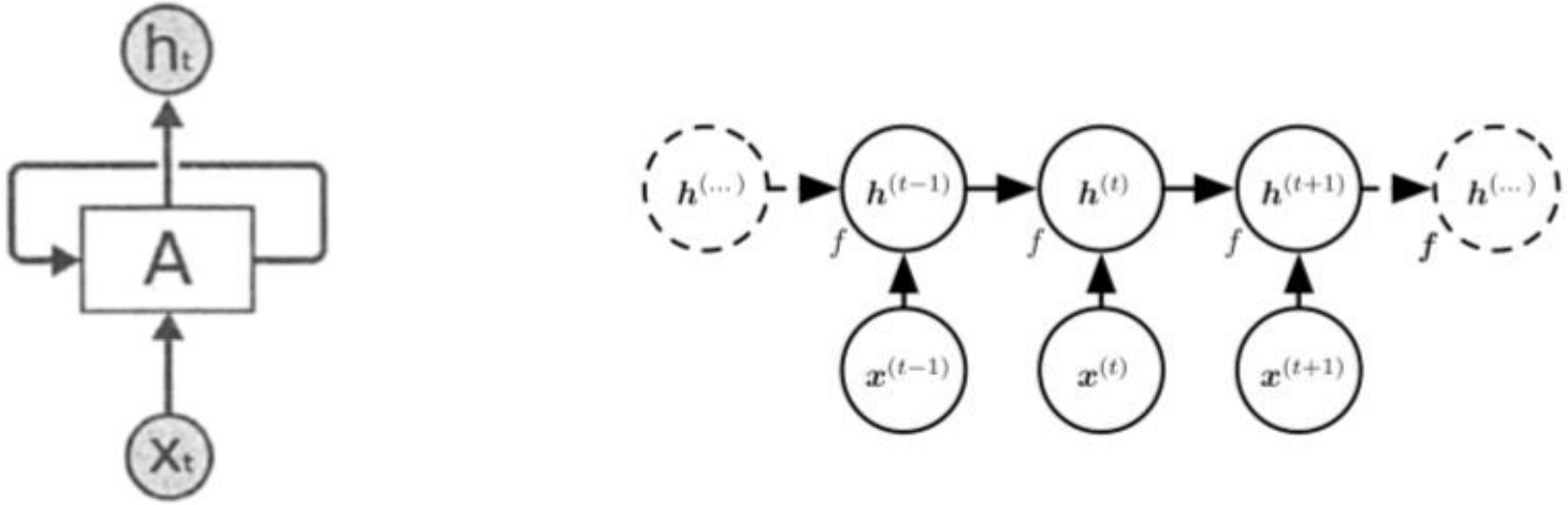
第三次卷积可以提取出高层次的特征。

特征是不断进行提取和压缩的，最终能得到比较高层次特征。简言之就是对原式特征一步又一步的浓缩，最终得到的特征更可靠。利用最后一层特征可以做各种任务：比如分类、回归等。

循环神经网络（RNN）

基本概念

循环神经网络的主体结构A的输入除了来自输入层**Xt**，还有一个循环的边来提供上一时刻的隐藏状态**ht−1**。在每一时刻，循环神经网络的模块A在读取了**Xt**和**ht−1**之后会生成新的隐藏状态**ht**，并产生本时刻的输出**O**t。循环神经网络当前的状态**ht**是根据上一时刻的状态**ht−1**和当前的输入**Xt**共同决定的。

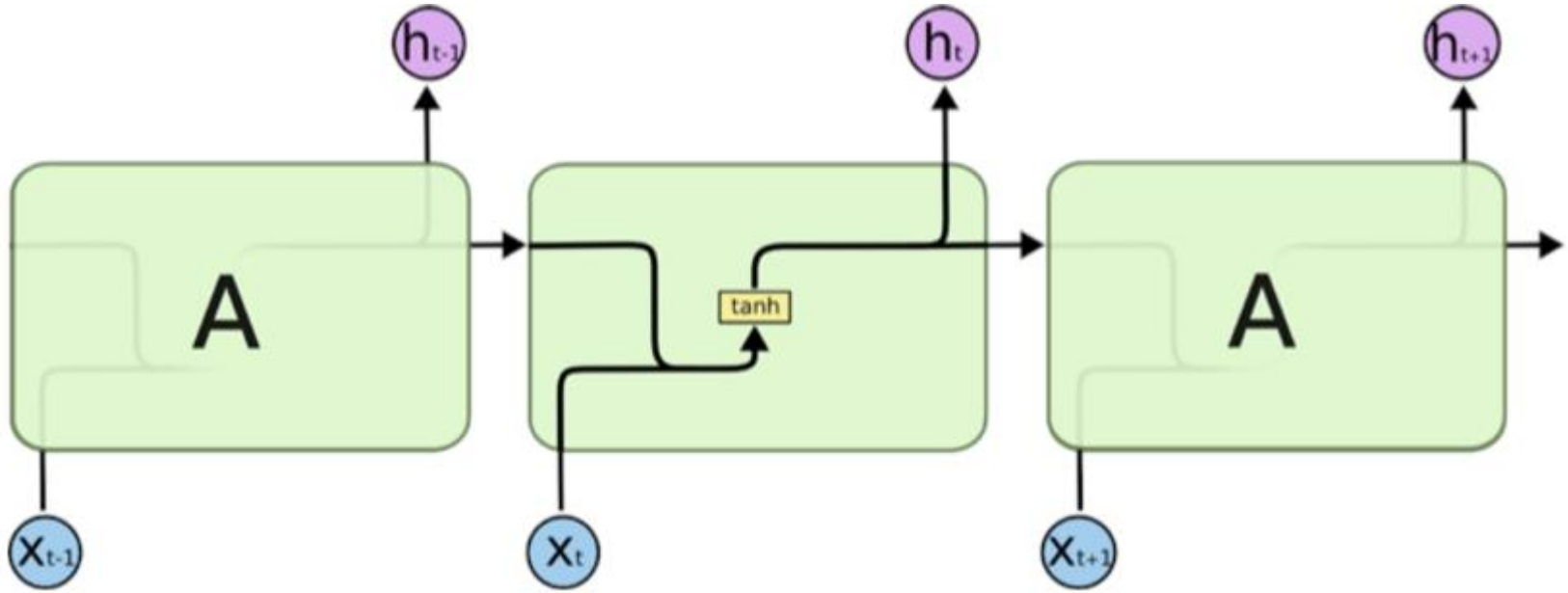


循环神经网络示意图

循环神经网络（RNN）

简单循环神经网络:

普通RNN



使用单层全连接神经网络作为循环体的RNN结构图

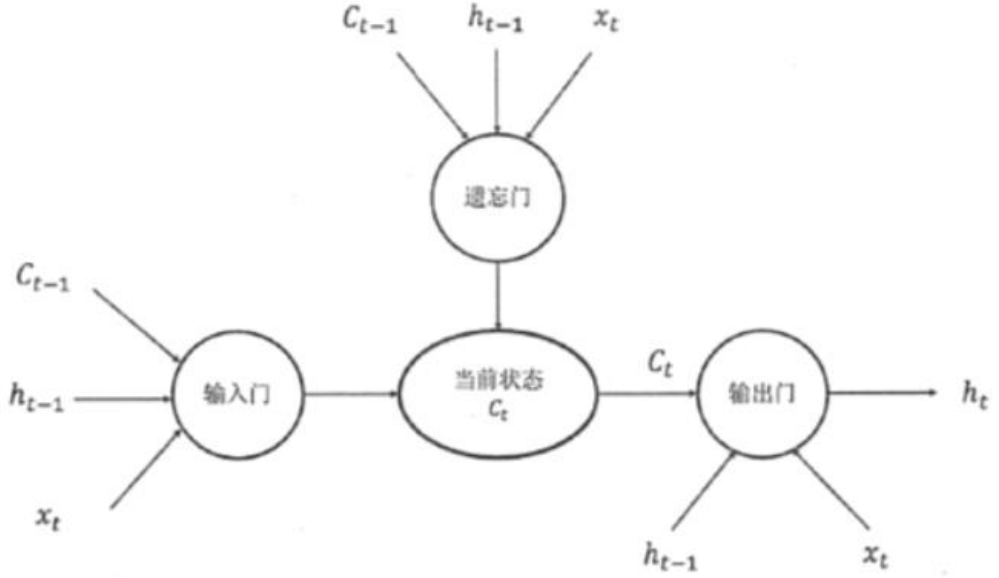
循环神经网络（RNN）

LSTM

长短时记忆模型:

当预测位置和相关信息之间的文本间隔不断增大时，简单循环神经网络有可能会丧失学习到距离如此远的信息的能力，或者在复杂语言场景中，有用信息的间隔有大有小、长短不一，循环神经网络的性能也会受到限制。

采用LSTM结构的循环神经网络比标准的循环神经网络表现更好。与单一tanh循环体结构不同，LSTM是一种拥有三个“门”结构的特殊网络结构。



- “门”的结构让信息有选择性地影响循环神经网络中每个时刻的状态。所谓“门”结构就是一个使用sigmoid神经网络和一个按位做乘法的操作。
- 使用sigmoid作为激活函数的全连接神经网络层会输出一个0到1之间的数值，描述当前输入有多少信息量可以通过这个结构。
- 当门打开时（sigmoid神经网络层输出为1时），全部信息都可以通过；当门关上时（sigmoid神经网络层输出为0时），任何信息都无法通过。

参考书籍推荐：

Referenced Textbook