机器学习期末

第二章 线性模型

2.1 线性回归

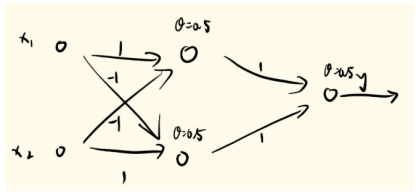
一元线性回归

$$w=\frac{\sum_{i=1}^{m}(x^{(i)}-\bar{x})(y^{(i)}-\bar{y})}{\sum_{i=1}^{m}(x^{(i)}-\bar{x})^2},\quad b=\bar{y}-w\bar{x}$$

第三章 感知机与神经网络

3.1. 感知机

设计一个两层感知机用于解决异或问题



对训练样例 $(\boldsymbol{x},y)$，若当前输出为 $\hat{y}$，则按如下方式调整权重：

$$w_i\leftarrow w_i+\Delta w_i,\Delta w_i=\eta(y-\hat{y})x_i$$

第四章 支持向量机

支持向量 距离超平面最近的样本点

间隔 两个异类支持向量到超平面的距离之和 $\gamma=\frac{2}{\|\boldsymbol{w}\|}$

划分超平面 $\boldsymbol{w}^T\boldsymbol{x}+b=0$ 即找到 $\boldsymbol{w}$ 和 b 使得间隔最大，等价于以下约束最值问题：

$$\min_{\boldsymbol{w},b}\ \frac{1}{2}\|\boldsymbol{w}\|^2,\quad s.t.\ y_i(\boldsymbol{w}^T\boldsymbol{x_i}+b)\geq 1$$

第五章 贝叶斯分类

符号定义

假设有 K 种可能的类别标记 $y=\{c_1,c_2,...,c_K\}$

输入为 N 个样本 $D=\{(x_1,y_1),(x_1,y_2),...,(x_N,y_N)\}$

样本有 n 维特征：$x_i=\left(x_i^{(1)},x_i^{(2)},...,x_i^{(n)}\right)$

第 j 维可能的取值有 S_j 种：$x^{(j)}\in\left\{a_{j1},a_{j2},...,a_{jS_j}\right\}$

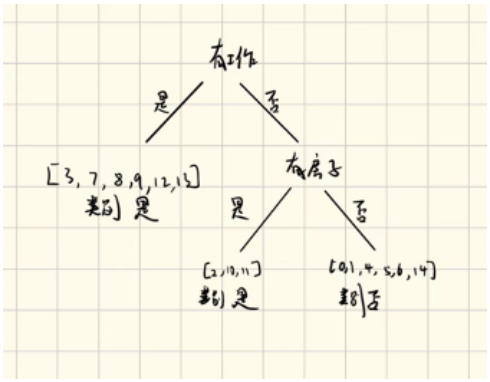
计算方式

1. 计算所有的 $P(Y=c_i),\ i=1,...,K$
2. 对于每个 c_i 计算所有的条件概率 $P\left(X^{(j)}=a_{jk}\mid Y=c_i\right),k=1,...,S_j$
3. 对于样本 $x=\left(x^{(1)},x^{(2)},...,x^{(n)}\right)$，对每个 c_i 计算：$P(Y=c_i)\prod_{k=1}^{n}P\left(X^{(k)}=x^{(k)}|Y=c_i\right)$
4. 最大的那个即为最终分类。

第六章 决策树

6.1 CLS 算法

通过依次选取特征分裂节点构建决策树：



6.2 ID3 算法

使用 **信息增益** 指导特征的选择过程。

事件 a_i 的信息熵：

$$H(a_i)=-p(a_i)\log_2 p(a_i)$$

对于随机变量 X, 若 $p_i=P(X=x_i)$，则此随机变量的信息熵：

$$H(X)=-\sum_i^n p_i\log_2 p_i,\quad p_i=P(X=x_i)$$

选取某一特征 A 所产生的 **信息增益** 即 D 的信息熵在“得知 A 的各个取值情况下的信息”的条件下，其信息熵减少了多少：

$$g(D,A)=H(D)-H(D|A)$$

此处引入条件熵，设 A 有 m 种取值 $a_1,a_2,...,a_m$ 则上式中的条件熵为：

$$H(D|A)=\sum_i^m P(A=a_i)\cdot H(D|A=a_i)$$

每次选取信息增益最大的特征来构建决策节点即可。

6.3 C4.5 算法

用 **信息增益率** 取代 **信息增益**（其实是做了个归一化）：

$$g_{R(D,A)}=\frac{g(D,A)}{H(A)}$$

第九章 降维

KNN 监督学习，懒惰学习

懒惰学习”(lazy learning) 此类学习技术在训练阶段仅仅是把样本保存起来，训练时间开销为零，待收到测试样本后再进行处理。

急切学习”(eager learning) 在训练阶段就对样本进行学习处理的方法。

9.1 主成分分析

即旋转坐标轴找到方差最大的方向作为新的坐标，并将数据投影到该坐标轴上。

以二维数据为例：

$$X=\begin{bmatrix}2&3&3&4&5&7\\2&4&5&5&6&8\end{bmatrix}$$

首先对其进行标准化：$x_{ij}=\frac{x_{ij}-\bar{x_i}}{\sqrt{s_{ii}}}$

$$\bar{X}=\begin{bmatrix}4\\5\end{bmatrix},s_{11}=3.2,s_{22}=4\sqrt{s_{11}}=\frac{4}{\sqrt{5}},\sqrt{s_{22}}=2$$

$$X^*=\begin{bmatrix}-\frac{1}{2}\sqrt{5}&-\frac{\sqrt{5}}{4}&-\frac{\sqrt{5}}{4}&0&-\frac{\sqrt{5}}{4}&\frac{3}{4}\sqrt{5}\\-\frac{3}{2}&-\frac{1}{2}&0&0&\frac{1}{2}&\frac{3}{2}\end{bmatrix}$$

然后计算协方差阵 $R=\frac{X^*X^{*^T}}{n-1}$：

$$R = \frac{X^* X^{*^T}}{5} = \begin{bmatrix} 1 & 17\frac{\sqrt{5}}{40} \\ 17\frac{\sqrt{5}}{40} & 1 \end{bmatrix}$$

求解 $|R - \lambda I| = 0$ 得到 k 个特征值及单位特征向量：

$\lambda_1 = 1 + 17\frac{\sqrt{5}}{40}, \lambda_2 = 1 - 17\frac{\sqrt{5}}{40}, \alpha_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$

计算主成分：$y_i = \alpha_i^T \boldsymbol{x}, i = 1, 2, ..., k$

于是有
$Y = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right) X^* = \left(-\frac{3+\sqrt{5}}{2\sqrt{2}}, \frac{\sqrt{5}+2}{4\sqrt{2}}, \frac{\sqrt{2}}{-4\sqrt{2}}, 0, \frac{\sqrt{5}+2}{4\sqrt{2}}, \frac{3\sqrt{5}+6}{4\sqrt{2}}\right)$

第八章 聚类

K-means 无监督，选取 k 个初始聚类中心，计算所有样本到各个聚类中心的距离，归入最近的类别；重新用类别内样本坐标均值计算聚类中心，进行迭代，直至聚类中心不再变化。

硬聚类 一个样本只能属于一个簇，或簇的交集为空集

软聚类 一个样本可以属于多个簇，或簇的交集不为空集

原型聚类 先对原型进行初始化，再对原型进行迭代更新求解 k 均值、学习向量量化算法、高斯混合聚类算法

密度聚类 从样本密度的角度考察样本的连接性，使密度相连的样本归结到一个簇，更符合直观认知 DBSCAN(Density-Based Spatial Clustering of Applications with Noise)

层次聚类 假设簇之间存在层次结构，将样本聚到层次化的簇中 聚合聚类（自下而上）、分裂聚类（自上而下）为硬聚类

第七章 集成学习

集成学习 构建并集成多个“好而不同”的学习器来完成学习任务

有如下两类：

7.1 Bagging 与随机森林

Bagging（Bootstrap aggregating，自聚汇聚算法） 从训练集中进行子抽样组成每个基模型所需要的子训练集，对所有基模型预测的结果进行综合，产生最终的预测结果

1. 通过降低基学习器的方差，改善了泛化误差；
2. 性能依赖于基学习器的稳定性；
3. 由于每个样本被选中的概率相同，因此 Bagging 并不侧重于训练数据集中的任何特定实例；
4. 时间复杂度低：假定基学习器的计算复杂度为 O(m)，采样与投票/平均过程的复杂度为 O(s)，则 Bagging 的复杂度大致为 T(O(m)+O(s))。

随机森林 用随机的方式建立一个森林。随机森林算法由很多决策树组成，每一棵决策树之间没有关联。建立完森林后，当有新样本进入时，每棵决策树都会分别进行判断，然后基于投票法给出分类结果。是 Bagging 的扩展变体，它在以决策树为基学习器构建 Bagging 集成的基础上，进一步在决策树的训练过程中引入了随机特征选择。随机选择样本和 Bagging 相同，采用的是 Bootstraping 自助采样法；随机选择特征是 指在每个节点在分裂过程中都是随机选择特 征的（区别与每棵树随机选择一批特征）

1. 在数据集上表现良好，相对于其他算法有较大的优势
2. 易于并行化，在大数据集上有很大的优势；
3. 能够处理高维度数据，不用做特征选择

7.2 Boosting

Boosting 训练过程为阶梯状；基模型按次序一一进行训练（实现上可以做到并行）；基模型的训练集按照某种策略每次都进行一定的转化；对所有基模型预测的结果进行线性综合产生最终的预测结果。

Stacking 将训练好的所有基模型对训练集进行预测，第 j 个基模型对第 i 个训练样本的预测值将作为新的训练集中第 i 个样本的第 j 个特征值，最后基于新的训练集进行训练。

简述 AdaBoost 和 GBDT 之间的联系和区别。

AdaBoost 和 GBDT 都是集成学习方法，通过组合多个弱学习器来构建一个强学习器。 它们都属于迭代的集成方法，每一轮迭代都会调整样本的权重，使得之前的错误更加受到关注。

他们的损失函数不同，AdaBoost 的损失函数主要关注错误分类的样本，通过提高错误样本的权重来改善分类效果。而 GBDT 使用的是梯度提升算法，每一轮迭代都会拟合残差，以减小模型的残差。GBDT 的损失函数通常是平方损失或者绝对损失。

他们的基学习器也不同，AdaBoost 每一轮迭代都会根据样本权重训练一个弱学习器，并将其加权组合。而 GBDT 每一轮迭代都训练一个新的弱学习器来拟合前面轮次的模型的残差。

他们权重更新方式也不同，AdaBoost 增加错误样本的权重，而 GBDT 通过拟合残差来更新样本的权重。

比较支持向量机、AdaBoost、逻辑斯谛回归模型的学习策略与算法。

SVM 采用结构风险最小化的思想，通过最大化分类边界两侧的间隔，使得模型对未见数据具有较好的泛化能力。其算法核心是寻找一个超平面，使得离该超平面最近的样本点到该超平面的距离（间隔）最大。

AdaBoost 通过组合多个弱学习器，每一轮迭代都关注之前模型分类错误的样本，以提高整体模型的性能。其算法核心是在每轮迭代中训练一个弱学习器，样本的权重会根据之前的分类结果进行调整，使得分类错误的样本在下一轮更受关注。

逻辑斯谛回归通过极大似然估计来估计模型参数，最大化观测数据的似然函数，从而找到最有可能产生观测数据的模型参数。 其算法核心是使用逻辑斯谛函数（sigmoid 函数）来建模二分类问题的概率分布。模型的参数通过梯度下降等优化算法来学习。