QG 工作室数据挖掘小组实验报告

实习生： 许继元　　导师：　　日期： 2020 年 08 月 06 日

实验名称：数据挖掘理论学习第二阶段
已完成内容： 1. BPNN 模型 2. KNN 算法 3. NaiveBayes 算法 4. K-Means 算法 5. GBDT 模型
未完成内容：暂无
未完成原因：暂无
需要帮助：暂无

实验总结

知识点总结：

BPNN 模型：

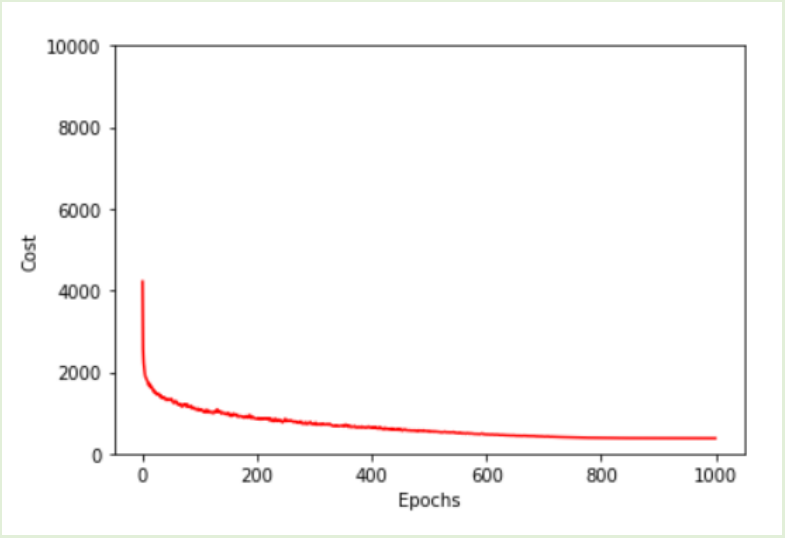
简介：

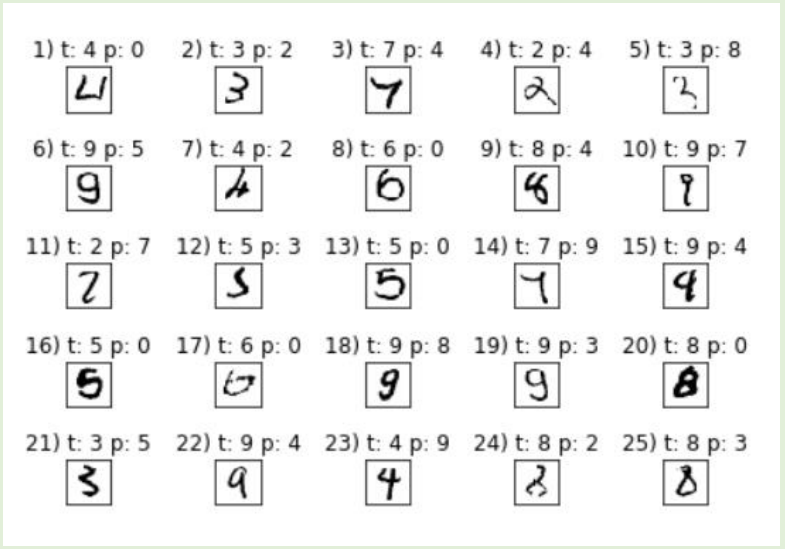
BP 算法是训练神经网络的常见方法，其对网络中所有权重计算损失函数的梯度，该梯度会反馈给最优化方法，用来更新权值以最小化损失函数。 反向传播要求有对每个输入值期望得到的已知输出，来计算损失函数的梯度。

流程：

1. 对每个训练样例，先将其提供给输入层神经元，然后逐层将信号前传，直到产生输出层的结果；
2. 计算输出层的误差，再将误差逆向传播至隐层神经元；
3. 根据隐层神经元的误差来对连接权和阈值进行调整；
4. 1、2、3 过程迭代进行，直到达到停止条件为止。

实践：在手写字体数据集上测试了 BPNN 模型。





训练准确率：97.59%

测试准确率：95.86%

BPNN 模型优点：

• 具有实现任何复杂非线性映射的功能。这使得它特别适合于求解内部机制复杂的问题；

• 具有自学习能力；

• 具有一定的推广、概括能力；

BPNN 模型优点：

• 学习速度很慢；

• 网络结构的选择一般只能由经验选定；

• 容易出现“过拟合”现象；

KNN 算法：

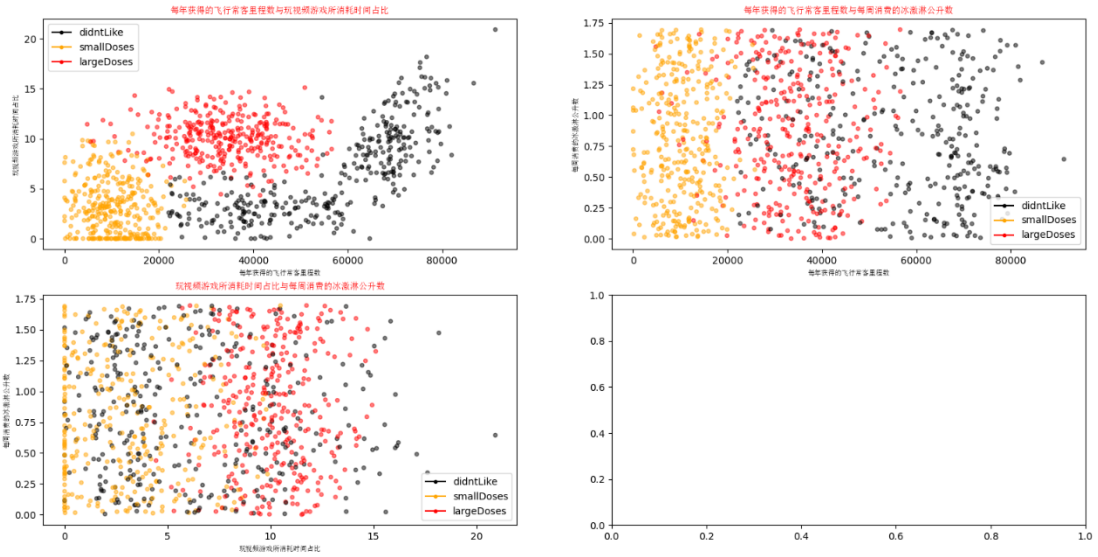
简介：

KNN 是一种常用的监督学习方法，其工作机制非常简单：给定测试样本，基于某种距离度量找出训练集中与其最靠近的 k 个训练样本，然后基于这 k 个“邻居”的信息来进行预测。

流程：

1. 计算测试数据与各个训练数据之间的距离；
2. 对距离从小到大进行排序；
3. 选取距离最小的 k 个点；
4. 确定前 k 个点类别的出现频率；
5. 出现频率最高的类别作为预测分类；

实践：在约会数据集上测试了 KNN 算法。



KNN 算法优点：

- 简单好用，容易理解，精度高，理论成熟，既可以用来做分类也可以用来做回归；
- 可用于数值型数据和离散型数据；
- 训练时间复杂度为 O(n)；
- 无数据输入假定；
- 对异常值不敏感。

KNN 算法缺点：

- 计算复杂性高；空间复杂性高；
- 样本不平衡问题；
- 一般数值很大的时候不用这个，计算量太大。但是单个样本又不能太少，否则容易发生误分。
- 最大的缺点是无法给出数据的内在含义。

NaiveBayes 算法：

简介：

NaiveBayes 算法以贝叶斯定理为基础，是贝叶斯分类中最简单，也是常见的一种分类方法。其采用了“属性条件独立性假设”：对已知类别，假设所有属性相互独立。

流程：

1. 导入数据集，确定特征属性；
2. 对每个类别计算$\boldsymbol{P(y_i)}$；
3. 对每个特征属性计算所有划分的条件概率；
4. 对每个类别计算$\boldsymbol{P(x \mid y_i)P(y_i)}$；
5. 以$P(x \mid y_i)P(y_i)$最大项作为x所属类别

实践：在垃圾邮件分类数据集中进行了算法测试。

分类错误率：10.00%

NaiveBayes 算法优点：

- 朴素贝叶斯模型发源于古典数学理论，有着坚实的数学基础，以及稳定的分类效率；
- 对大数量训练和查询时具有较高的速度；
- 对小规模的数据表现很好，能个处理多分类任务，适合增量式训练；
- 对缺失数据不太敏感，算法也比较简单，常用于文本分类；
- 朴素贝叶斯对结果解释容易理解；

NaiveBayes 算法缺点：

- 需要计算先验概率；
- 分类决策存在错误率；
- 对输入数据的表达形式很敏感；
- 由于使用了样本属性独立性的假设，所以如果样本属性有关联时其效果不好。

K-Means 算法：

简介：

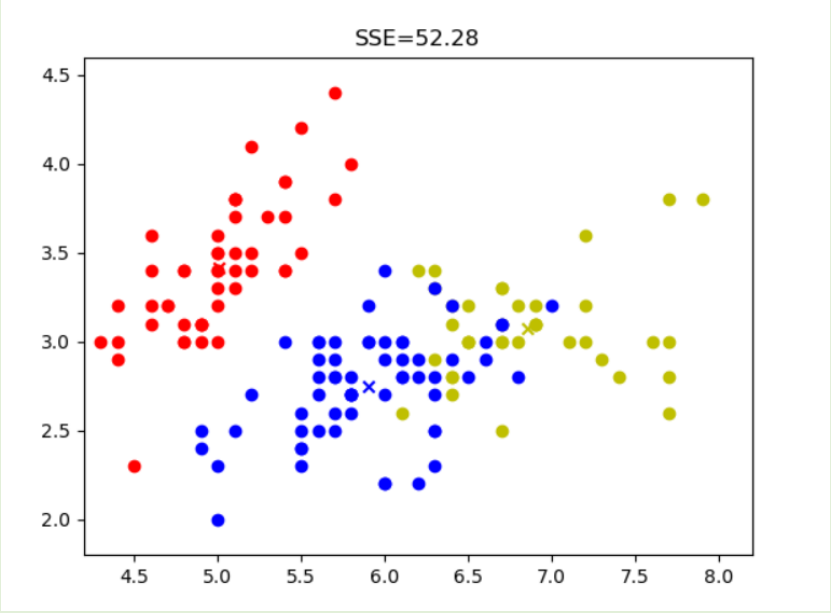
K-Means 算法是一种广泛使用的聚类算法，其主要思想是通过迭代过程把数据集划分为不同的类别，使得评价聚类性能的准则函数达到最优，从而使生成的每个聚类内紧凑，类间独立。K-Means 算法不适合处理离散型属性，但是对于连续型具有较好的聚类效果。

流程：

1. 随机选取 K 个初始点为质心（类别）；
2. 遍历每一条数据，计算其与 K 个质心的距离；
3. 选择与之距离最近的质心作为该数据所属的类别；
4. 每个簇的质心更新为该簇所有点的平均值；
5. 重复 2、3、4 步骤，直到代价函数收敛到最小值；

实践：

在 iris 数据集上测试了 K-Means 算法。



K-Means 算法优点：

• 容易实现；

• 原理简单，收敛速度快；

• 处理大数据集较为高效。空间复杂度为 O(N)，时间复杂度为 O(IKN)——N 为样本点个数，K 为中心点个数，I 为迭代次数；

• 需要调参的只有簇数 K

K-Means 算法缺点：

• 对初值敏感，对于不同的初始值，可能会导致不同结果；

• 在簇的平均值被定义的情况下才能使用，这对于处理符号属性的数据不适用；

• 对于不是凸的数据集比较难收敛；

• 对于不平衡的数据，聚类效果不佳；

• 对噪音和异常点比较敏感；

GBDT 模型：

简介：

GBDT 模型是机器学习领域中浅层模型的优秀模型，也是各大数据挖掘比赛中经常出现的框架。其全称是 GradientBoostingDecisionTree，中文名是梯度提升树。

GBDT 模型的损失函数：

GBDT 模型的基模型为 DT（决策树），即对于 GBDT 下的每轮迭代，输出的$f_m(x)$为决策树。而最终的 GBDT 模型为所有决策树输出结果之和。Boosting 的基模型采用的都是弱模型，因此，通常 GBDT 基模型的决策树树深不会太深（一般少于 5 层）。

回归决策树的损失函数采用平方误差，GBDT 也可以保持一致。平方误差的公式为：

$$\boldsymbol{L(w)=\sum_{i=1}^{n}(\hat{y}_i-y_i)^2}$$

此处的模型和损失函数都是参数 w 的函数，参数 w 为每个基模型的每个分裂特征和每个分裂阈值。

对于损失函数中的预测值$\boldsymbol{y_i}$有：

$$y_i=\sum_{m=1}^{M}f_m(x_i)=\sum_{m=1}^{M-1}f_m(x_i)+f_M(x_i)$$

代入上式，有：

$$L(w)=\sum_{i=1}^{n}(\hat{y}_i-y_i)^2=\sum_{i=1}^{n}\left(\hat{y}_i-\sum_{m=1}^{M-1}f_m(x_i)-f_M(x_i)\right)^2$$

观察上式，发现损失函数中有 3 项。第一项是真实值，第二项是截止到第 M-1 棵树的预测值，第三项是第 M 棵树的输出结果，若令：

$$\boldsymbol{r_i=\hat{y}_i-\sum_{m=1}^{M-1}f_m(x_i)}$$

表示的是，在训练第 M 棵树时，截止到当前的残差值（误差），将其代入上式，有：

$$L(w)=\sum_{i=1}^{n}\left(r_i-f_M(x_i)\right)^2$$

该公式表明，为了让损失函数数值最小，在训练某个基模型时，其训练目标是去拟合残差$\boldsymbol{r_i}$。

GBDT 模型的最优化求解：

上述讨论是基于损失函数为平方误差的情况。一般地，如果损失函数不是平方误差，则每个基模型的训练目标就不是残差。GBDT 利用损失函数的负梯度在当前模型的值作为残差的近似值，即：

$$r_i \approx -\left[\frac{\partial L(w)}{\partial f_{M-1}(x)}\right]$$

特别地，当损失函数采用平方误差时，损失函数的负梯度就是先前推导的残差：

$$-\left[\frac{\partial L(w)}{\partial f_{M-1}(x)}\right] = \hat{y}_i - \sum_{m=1}^{M-1} f_m(x_i)$$

GBDT 模型的优点：

- 预测精度高；
- 适合低维数据；
- 能处理非线性数据；
- 可以灵活处理各种类型的数据，包括连续值和离散值；
- 在相对少的调参时间情况下，预测的准备率也可以比较高；
- 使用一些健壮的损失函数，对异常值的鲁棒性非常强（如 Huber 损失函数和 Quantile 损失函数）

GBDT 模型的缺点：

- 由于弱学习器之间存在依赖关系，难以并行训练数据。不过可以通过自采样的 SGBT 来达到部分并行；
- 如果数据维度较高时会加大算法的计算复杂度；

遇到问题： 部分机器学习算法大量涉及概率论的知识，在推导过程中许多符号和公式都不懂，比较吃力。	解决过程： 边学边查阅资料，参考网上相关博客的讲解。

导师评价				
实验分数	知识掌握情况	代码编写能力	建议	评价日期