5.7 监督学习方法

5.7.1 概率监督学习

可以使用最大似然估计找到对于有参分布族p(y|x; θ)最好的参数向量θ。

通过定义一族不同的概率分布，我们可以将线性回归扩展到分类情况中

线性回归的实数正态分布是用均值参数化的

使用 logistic sigmoid 函数将线性函数的输出压缩进区间(0,1)

$$p(y = 1 \mid \boldsymbol{x}; \boldsymbol{\theta}) = \sigma(\boldsymbol{\theta}^\top \boldsymbol{x}).$$

这个方法被称为逻辑回归，用于分类

5.7.2 支持向量机

类似于逻辑回归，但是不输出概率，只输出类别（正类和负类）

核技巧：

重写线性函数

$$\boldsymbol{w}^\top \boldsymbol{x} + b = b + \sum_{i=1}^{m} \alpha_i \boldsymbol{x}^\top \boldsymbol{x}^{(i)},$$

点积替换为核函数

$$k(\boldsymbol{x}, \boldsymbol{x}^{(i)}) = \phi(\boldsymbol{x}) \cdot \phi(\boldsymbol{x}^{(i)})$$

之后，我们可以使用以下函数进行预测：

$$f(\boldsymbol{x}) = b + \sum_i \alpha_i k(\boldsymbol{x}, \boldsymbol{x}^{(i)}).$$

核技巧十分强大：

它使我们能够使用保证有效收敛的凸优化技术来学习非线性模型

核函数 k 的实现方法通常有比直接构建 φ(x) 再算点积高效很多

最常用的核函数是高斯核

$$k(\boldsymbol{u}, \boldsymbol{v}) = \mathcal{N}(\boldsymbol{u} - \boldsymbol{v}; \boldsymbol{0}, \sigma^2 \boldsymbol{I})$$

核机器的一个主要缺点：

计算决策函数的成本关于训练样本的数目是线性的

5.7.3 其他简单的监督学习算法

k-最近邻算法：

k-最近邻算法没有任何参数，而是使用训练数据的简单函数

甚至也没有一个真正的训练阶段或学习过程

几乎适用于任何类型可以确定 y 值平均值的监督学习

k-最近邻的高容量使其在训练样本数目大时能够获取较高的精度

缺点：

k-最近邻的一个弱点是它不能学习出哪一个特征比其他更具识别力

决策树：

决策树的每个节点都与输入空间的一个区域相关联
内部节点继续将区域分成子节点下的子区域
空间由此细分成不重叠的区域，叶节点和输入区域之间形成一一对应的关系
缺点：
由于决策树通常使用坐标轴相关的拆分，并且每个子节点关联到常数输出，因此有时解决一些对于逻辑回归很简单的问题很费力

5.8 无监督学习算法

无监督算法只处理“特征”，不操作监督信号
一个经典的无监督学习任务是找到数据的“最佳”表示
该表示在比本身表示的信息更简单或更易访问
受到一些惩罚或限制的情况下，尽可能地保存关于 x 更多的信息
最常见的三种：
低维表示
稀疏表示
独立表示

5.8.1 主成分分析

假设有一个 m* n 的设计矩阵 X，数据的均值为零（数据可以很容易地中心化）
X 对应的无偏样本协方差矩阵：

$$\mathrm{Var}[\boldsymbol{x}] = \frac{1}{m-1}\boldsymbol{X}^\top \boldsymbol{X}$$

PCA 通过线性变换找到一个 Var[z] 是对角矩阵的表示

$$\boldsymbol{z} = \boldsymbol{W}^\top \boldsymbol{x}$$

我们有

$$\boldsymbol{X}^\top \boldsymbol{X} = \boldsymbol{W}\boldsymbol{\Lambda}\boldsymbol{W}^\top$$

主成分也可以通过奇异值分解得到
假设 W 是奇异值分解$\boldsymbol{X} = \mathrm{U}\Sigma W^T$的右奇异向量
以 W 作为特征向量基：

$$\boldsymbol{X}^\top \boldsymbol{X} = \left(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{W}^\top\right)^\top \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{W}^\top = \boldsymbol{W}\boldsymbol{\Sigma}^2\boldsymbol{W}^\top$$

PCA 后的 Var[z] 是对角的：

$$\begin{aligned}
\mathrm{Var}[\boldsymbol{z}] &= \frac{1}{m-1}\boldsymbol{Z}^\top \boldsymbol{Z} \\
&= \frac{1}{m-1}\boldsymbol{W}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{W} \\
&= \frac{1}{m-1}\boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{\Sigma}^2\boldsymbol{W}^\top \boldsymbol{W} \\
&= \frac{1}{m-1}\boldsymbol{\Sigma}^2,
\end{aligned}$$

所以，z 中的元素是彼此无关的

PCA 将数据变换为元素之间彼此不相关表示的能力是 PCA 的一个重要性质

5.8.2 k-均值聚类

k-均值聚类算法将训练集分成 k 个靠近彼此的不同样本聚类

提供的 one-hot 编码也是一种稀疏表示

自然地传达了相同聚类中的样本彼此相似的观点

具有计算上的优势

但是丢失了很多分布式表示的优点

k-均值聚类初始化 k 个不同的中心点$\{\mu^{(1)}, \ldots, \mu^{(k)}\}$，之后迭代直至收敛：

步骤一，每个训练样本分配到最近的中心点$\mu^{(i)}$所代表的聚类 i

步骤二，每一个中心点$\mu^{(i)}$更新为聚类 i 中所有训练样本$x^{(j)}$ 的均值

聚类问题本身是病态

没有单一的标准去度量聚类的数据在真实世界中效果如何

5.9 随机梯度下降

好的泛化需要大的训练集，但大的训练集的计算代价也更大

例如

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{x},\mathrm{y}\sim\hat{p}_{\mathrm{data}}} L(\boldsymbol{x}, y, \boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} L(\boldsymbol{x}^{(i)}, y^{(i)}, \boldsymbol{\theta})$$

$$L(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\log p(y \mid \boldsymbol{x}; \boldsymbol{\theta})$$

此时，梯度下降需要计算：

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}} L(\boldsymbol{x}^{(i)}, y^{(i)}, \boldsymbol{\theta})$$

这个运算的计算代价是 O(m)，对于数十亿的样本，计算一步梯度的耗时都会非常大

随机梯度下降

核心是，梯度是期望

期望可使用小规模的样本近似估计

每次更新计算只用到几百个样本

梯度的估计：

$$\boldsymbol{g} = \frac{1}{m'}\nabla_{\boldsymbol{\theta}} \sum_{i=1}^{m'} L(\boldsymbol{x}^{(i)}, y^{(i)}, \boldsymbol{\theta})$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \epsilon \boldsymbol{g}.$$

随机梯度下降在深度学习之外有很多重要的应用

每一步随机梯度下降更新的计算量不取决于训练集的大小 m

达到收敛所需的更新次数通常会随训练集规模增大而增加

但是该模型最终会在抽样完训练集上的所有样本之前收敛到最优

5.10 构建机器学习算法

几乎所有的深度学习算法都可以被描述为一个相当简单的配方：

特定的数据集

代价函数

优化过程

模型

可以替换独立于其他组件的大多数组件，得到不同的算法

组合模型、代价和优化算法来构建学习算法的配方同时适用于监督学习和无监督学习。

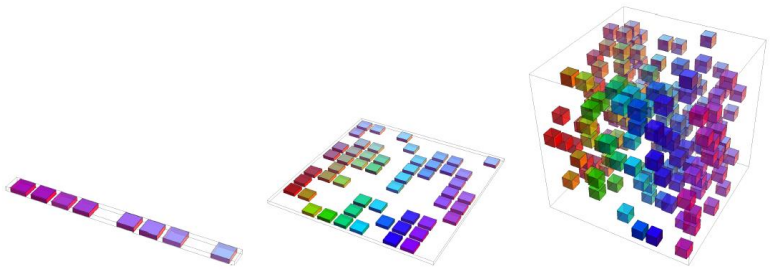
5.11 促使深度学习发展的挑战

传统学习算法在诸如语音识别或者对象识别问题上泛化能力不足

5.11.1 维数灾难

维数灾难带来的一个挑战是统计挑战

x 的可能配置数目远大于训练样本的数目



在高维空间中参数配置数目远大于样本数目，大部分单元格中没有样本

5.11.2 局部不变性和平滑正则化

最广泛使用的隐式“先验”是**平滑先验**或者**局部不变先验**

鼓励学习过程能够学习出函数f^*

然而，依赖此先验的结果是不能推广去解决人工智能级别任务中的统计挑战

如何引入额外的先验：

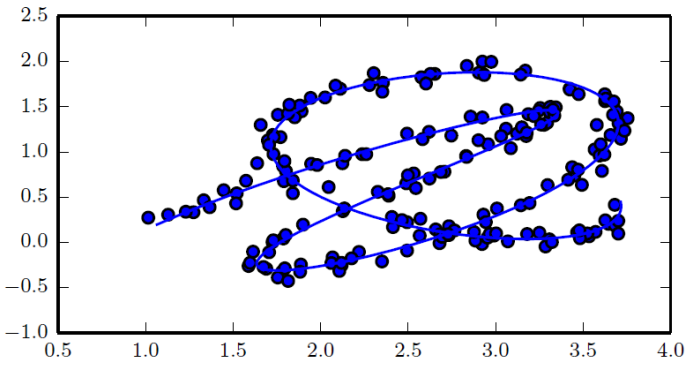
神经网络不会包含这些很强的（针对特定任务的）假设

深度学习的核心思想是假设数据由因素或特征组合产生

因素或特征可能来自一个层次结构的多个层级

5.11.3 流形学习

流形（manifold）指连接在一起的区域



训练数据位于二维空间中的一维流形中

流形学习假设认为$\mathbb{R}^n$中大部分区域都是无效的输入，有意义的输入只分布在包含少量数据点的子集构成的一组流形中

学习函数的输出中，有意义的变化都沿着流形的方向或仅发生在我们切换到另一流形时

关键假设仍然是概率质量高度集中

支持流形假设的观点：

现实生活的很多领域中，均匀的噪声从来不会与结构化输入类似

均匀地随机抽取字母来生成文件，得到有意义英语文档的概率几乎为零

我们至少能够非正式地想象这些邻域和变换

人脸图像的流形不太可能连接到猫脸图像的流形

当数据位于低维流形中时，使用流形中的坐标而非$\mathbb{R}^n$中的坐标表示机器学习数据更为自然

道路是嵌入在三维空间的一维流形，用一维道路中的地址号码确定地址