Machine Learning Lecture Notes

yujie6@sjtu.edu.cn[1]

2020 年 3 月 29 日

[1]Yujie Lu, ACM class 18, ID is 518030910111

目录

第一章　Lecture 1 (Mar 2)

1.1　人工智能简介

1.1.1　什么是人工智能

智能是实现世界目标的计算能力部分。

人工智能探讨对机器进行设计的方法论使得其可以去完成**基于智能的任务**。

1.1.2　人工智能方法

基于规则的方法

- 直接编程实现
- 借鉴人类启发式学习

基于数据的方法

- 专家系统：基于数据创造决策的规则
- 机器学习：基于数据进行预测或决策

1.2　机器学习简介

1.2.1　机器学习定义

学习是**系统**通过**经验**提升性能的过程。

Definition 1.2.1 (Tom Mitchell) *机器学习是一门研究学习算法的学科，这些算法 (非显式编程) 能在某些任务 T 上通过经验 E 来提升性能 P。*

机器学习分类

- 预测
  - 监督学习

  - 无监督学习
- 决策
  - 在动态环境中采取行动（强化学习）

1.2.2 机器学习应用

预测

- 网页搜索（根据 profile 分析）
- 人脸识别（Computer vision）
- 推荐系统
- 在线广告
- 信息提取

决策

- 交互式内容推荐
- 机器人控制
- 自动驾驶
- 游戏智能
- 多智能体协作

1.2.3 机器学习基本思想

以监督学习为例，给定带 label 的数据集

$$D = \{(x_i, y_i)\}_{i=1,2,\cdots N}.$$

寻找 $\theta = (a, b, c, \cdots)$ 使得函数映射

$$y_i \approx f_\theta(x_i).$$

我们用 loss function

$$\mathcal{L}(y_i, f_\theta(x_i)) = \frac{1}{2}(y_i - f_\theta(x_i))^2$$

来衡量预测的误差

$$\mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y_i, f_\theta(x_i)).$$

通过梯度下降求得 $\min_\theta \mathcal{L}(\theta)$

模型选择

模型选得不好会导致**欠拟合**或**过拟合**。为了防止这两种情况，我们可以采用

正则化

添加 θ 的惩罚项 $\Omega(\theta)$，一般选择 L2 正则化 $\lambda\|\theta\|_2^2$（也称玲回归 Ridge）。当我们增加参数 θ_n 时，我们实际上是在学习前面的参数产生的**残差**

这样做的另一个解释是奥卡姆剃刀原则（Occam's Razor），这个原则是说能用简单的方法完成任务的就尽量不要复杂，在这里就是能用简单的模型去拟合就不用复杂的能把噪声都刻画出来的方法。

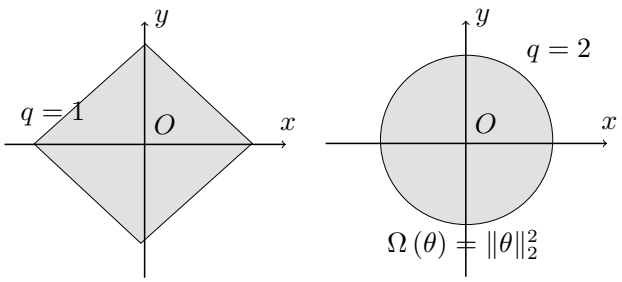

图 1.1: Ridge

有时候也会选择 $q \leq 1$ 进行稀疏性学习。

一个机器学习的解决方案的模型包含参数 θ 与超参数 λ。

Definition 1.2.2 *超参数为需要预先定义，无法直接从数据学习的参数。*

交叉验证

K-fold, 将训练集分成 k 份。

泛化能力

Definition 1.2.3 *泛化能力 (generalization ability) 指对未训练数据的预测能力。*

为了描述这种能力我们引入泛化误差 (Generalization Error):

$$R(f) = \int_{X\times Y} \mathcal{L}(y, f(x))\, p(x, y)\, \mathrm{d}x\, \mathrm{d}y.$$

其中 $p(x, y)$ 是潜在的联合数据分布 (joint probability distribution)。在已有数据集上也可进行经验估

计:

$$\hat{R}(f) = \frac{1}{N}\sum_{i=1}^{n}\mathcal{L}_i.$$

第二章　March 9

2.1　判别模型和生成模型

- 判别模型
  - 确定性判别：$y = f_\theta(x)$
  - 随机判别：$p_\theta(y \mid x)$
- 生成模型：建立联合概率分布（一般不用）

$$p_\theta(y \mid x) = \frac{p_\theta(x, y)}{p_\theta(x)}.$$

2.1.1　生成模型

- 频率派，θ 是具体的一个点，易于计算
- 贝叶斯派，θ 是一个分布

2.2　线性回归

一维的线性回归和二次回归都是线性模型：比如

$$f(x) = \theta_0 + \theta_1 x + \theta_2 x^2 = \theta^T \phi(x).$$

ϕ 实际上是一种 feature engineering。

学习目标

$$\mathcal{L}(\theta) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y_i, f_\theta(x_i)).$$

损失函数选择：min square loss

学习方法

梯度下降

$$\theta_{new} \leftarrow \theta_{old} - \eta \frac{\partial \mathcal{L}(\theta)}{\partial \theta}.$$

2.3 梯度下降

2.3.1 批量梯度下降

根据整个批量数据的梯度更新数据

$$\frac{\partial J(\theta)}{\partial \theta} = -\frac{1}{N} \sum_{i=1}^{N} (y_i - f_\theta(x_i)) x_i.$$

2.3.2 随机梯度下降

Stochastic Gradient Descent 在数据量较大时比批量梯度下降更为优秀。但是学习中存在**震荡**或不确定性。

优化目标

$$J^{(i)}(\theta) = \frac{1}{2} (y_i - f_\theta(x_i))^2.$$

2.3.3 小批量梯度下降

前面两种方式的结合，将训练集分为 K 个 mini-batch。对每一个小批量更新参数

$$J^{(k)}(\theta) = \frac{1}{2N_k} \sum_{i=1}^{N_k} (y_i - f_\theta(x_i))^2.$$

优点

- 结合 map-reduce 可以比较容易实现并行
- 更新速度快，不确定性低

2.3.4 基本搜索步骤

随机选择初始化参数，走到局部最优值。

Definition 2.3.1 $f: \mathbb{R}^n \to \mathbb{R}$ 是凸函数：$\mathrm{dom} f$[a] 是一个凸集，并且

$$f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t) f(x_2).$$

[a] dom 指函数定义域

而凸函数是一定有最值的。

2.4 从代数角度看线性回归

目标函数

$$J(\boldsymbol{\mu}) = \frac{1}{2}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu}).$$

而梯度为

$$\frac{\partial J(\boldsymbol{\mu})}{\partial \mu} = -\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu}).$$

甚至可以直接求出最优参数

$$\frac{\partial J(\boldsymbol{\mu})}{\partial \mu} = 0 \Rightarrow \hat{\mu} = \left(\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}.$$

但是当 $\boldsymbol{X}$ 很大时，这是很难计算的。

当 $\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X}$ 为奇异矩阵

其逆矩阵无法计算，解决方法

- Regulariazation
- $J_1(\mu) = J(\mu) + \frac{\lambda}{2}\|\mu\|_2^2$

从而

$$\hat{\mu} = \left(\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X} + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{y}.$$

2.4.1 泛线性模型

本质上是做一个替换 $\boldsymbol{x} \to \phi(\boldsymbol{x})$，$\phi(x)$ 是 $\mathbb{R}^d \to \mathbb{R}^h$ 的向量函数。加入了 ϕ 的线性回归也叫核线性回归。

2.4.2 核线性回归的矩阵形式

使用线性代数技巧得到：

$$\hat{y} = \Phi\hat{\theta} = \Phi\Phi^T\left(\Phi\Phi^T + \lambda I_n\right)^{-1}y.$$

只需关心核矩阵

$$K = \Phi\Phi^T = \{k\left(x^{(i)}, x^{(j)}\right)\}.$$

2.5 最大似然估计

带高斯白噪声的线性拟合:

$$y = f_\theta(x) + \varepsilon.$$

其中 $\varepsilon \sim \mathcal{N}(0, \sigma^2)$.

优化目标

最大似然 (Likelyhood).

$$p(y \mid x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{\varepsilon^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{y-\theta^T x}{2\sigma^2}}.$$

最大化这个似然

$$\max_\theta \prod_{i=1}^{N} p(y_i \mid x_i).$$

等价于最小均方误差的学习 (取对数即可证明)

2.6 分类指标

分类器有以下几个指标:

$$\begin{aligned} \text{Accuracy} &= \frac{TP + TN}{TP + FP + TN + FN} \\ \text{Precision} &= \frac{TP}{TP + FP} \\ \text{Recall} &= \frac{TP}{TP + FN}. \end{aligned}$$

为了权衡准确率和召回率，可以调整阀值 h:

$$\hat{y} = \begin{cases} 1, & p_\theta(y = 1 \mid x) > h \\ 0, & \text{else} \end{cases}.$$

为了判别分类器好不好，我们有 F1 score

$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

2.6.1 ROC 曲线

当 h 不断提升时，可以做出一条 $\frac{TP}{P} \sim \frac{FP}{N}$ 曲线 (Receiver operating characteristic). AUC[1] (Area under ROC curve) 可以用来衡量准确率，处于 0.75 以上时可以认为分类模型比较可靠。

[1] 一般处于 $(0.5, 1)$

第三章　March 16

3.1　逻辑回归

概率判别模型在分类时往往更加可靠，因为如果用函数的话就无法求导了。

3.1.1　二分类

损失函数用交叉熵来衡量

$$\mathcal{L}(y,x)=-\sum_k \delta(y=c_k)\log p_\theta(y=c_k\mid x).$$

其中

$$\delta(z)=\begin{cases}1, & z \text{ is true}\\ 0, & z \text{ else}\end{cases}.$$

为了输出一个 $(0,1)$ 之间的值，我们会给 $z=\theta^T x$ 套上一个 sigmoid 函数

$$\sigma(z)=\frac{1}{1+\exp(-z)}.$$

即

$$p(y=1\mid x)=\sigma(z),\quad p(y=0\mid x)=1-\sigma(z).$$

带入损失函数有

$$\mathcal{L}(y,x)=-y\log\sigma(z)-(1-y)\log(1-\sigma(z)).$$

有趣的是，σ 求导后

$$\frac{\partial\sigma(z)}{\partial z}=\sigma(z)(1-\sigma(z)).$$

从而梯度为

$$\frac{\partial \mathcal{L}(y,x)}{\partial \theta} = (\sigma(z) - y)\, x.$$

可以看到，梯度形式与线性回归十分相似。

3.1.2 多分类

与二分类同样的，只是引入一个类别集

$$C = \{c_1, c_2, \cdots, c_n\}.$$

使用 softmax

$$p_\theta(x = c_j \mid x) = \frac{\exp\left(\theta_j^T x\right)}{\sum_{i=1}^{m} \exp\left(\theta_i{}^T x\right)}.$$

参数 $\theta = \{\theta_1, \cdots, \theta_m\}$, 可以同时除掉 θ_1 来减少一个参数。

其实可以看成 m 个二分类，目标

$$\max \log p_\theta(y = c_j \mid x).$$

求梯度有

$$\begin{aligned}\frac{\partial \log p_\theta}{\partial \theta_j} &= x - p_\theta x \\ &= (1 - p_\theta(y = c_j \mid x))\, x.\end{aligned}$$

3.1.3 逻辑回归应用

在线广告点击率 (CTR) 估算 Feature engineering 十分重要

- One-Hot 二进制编码
- 例如`weekday=friday` 转化为一个七维向量。容易发现转化的高维向量极为稀疏。

3.2 支持向量机

线性分类器本质上是划一个决策边界，但是考虑到数据噪声后，分类器很容易出错。我们发现边界与数据距离最大时是最 robust 的，也称之为最大间隔边界。

我们的优化目标便是最短距离最大化。逻辑回归的打分函数 $s = \theta^T x$，容易发现如果打分越高的样例越远离决策边界，分类器越可靠。

3.2.1 SVM 最优化问题

假设

- 标签 $y \in \{-1, 1\}$
- $h(x) = g\left(w^T x + b\right)$ [1]

由高中知识我们知道，点到直线的距离即几何间隔为

$$\gamma^{(i)} = \frac{|w^T x^{(i)} + b|}{\|w\|^2}.$$

不妨设 $\|w\|^2 = 1$，且由于 $y^{(i)}$ 的取值只为 $\{-1, 1\}$，上式可简化为

$$\gamma^{(i)} = y^{(i)}\left(w^T x^{(i)} + b\right). \tag{3.1}$$

除此之外还可以通过简单的代数手段进行推导：

$$w^T\left(x^{(i)} - \gamma^{(i)} y^{(i)} \frac{w}{\|w\|}\right) + b = 0.$$

解得

$$\gamma^{(i)} = y^{(i)}\left[\left(\frac{w}{\|w\|}\right)^T x^{(i)} + \frac{b}{\|w\|}\right]. \tag{3.2}$$

容易发现(3.1)和(3.2)本质相同，从而最小几何间隔为

$$\hat{\gamma} = \min_i\left(\gamma^{(1)}, \cdots, \gamma^{(m)}\right).$$

等价于在

$$y^{(i)}\left(w^T x^{(i)} + b\right) \geq \hat{\gamma}.$$

时求 $\max \gamma$，但是非凸目标函数不易求最值，所以需要进行转换。将函数间隔固定为 1，即 $\hat{\gamma} = 1$，目标函数转化为

$$\min \frac{1}{2}\|w\|^2.$$

使得 $y^{(i)}\left(w^T x^{(i)} + b\right) \geq 1$

[1] g = sgn

3.3 支持向量机优化

3.3.1 拉格朗日对偶问题

对于凸优化问题，可以使用拉格朗日乘数法。但是只能处理限制条件是等式的情况，为了处理不等式，我们需要引入松弛变量将其转化为等式，例如

$$g_i(w) = 1 - y^{(i)}\left(w^T x^{(i)} + b\right) \leq 0.$$

$g_i(w)$ 即支持向量。这等价于存在 a_i 使得

$$g_i(w) + a_i^2 = 0.$$

得到拉格朗日函数

$$\mathcal{L}(w, \lambda, a) = f(w) + \sum_{i=1}^{n} \lambda_i \left(g_i(w) + a_i^2\right).$$

其中 f 即优化目标 $\frac{1}{2}\|w\|^2$ 由拉格朗日乘数法知

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial w} = 0 \\ \lambda_i a_i = 0 \\ g_i(w) + a_i^2 = 0 \\ \lambda_i \geq 0 \end{cases}.$$

对 $\lambda_i a_i = 0$ 讨论，若 $a_i = 0$ 则 $g_i(w) = 0$，若 $a_i \neq 0$ 则必有 $\lambda_i = 0$。

知 a_i 这个变量对我们求最值没有任何约束，我们可以用 $\lambda_i g_i(w) = 0$ 来代替上式的条件。删去 a_i 即可得到KKT 条件，

从而拉格朗日函数可以重写为

$$\mathcal{L}(w, \lambda) = f(w) + \sum_{i=1}^{n} \lambda_i g_i(w).$$

注意到 $\mathcal{L} \leq f(w)$ ，当调整 λ 使得 $\sum_{i=1}^{n} \lambda_i g_i(w)$ 大的时候，$f(w)$ 一定会更小，故我们的最优化问题等价于

$$\min_{w} \max_{\lambda} \mathcal{L}(w, \lambda).$$

可以证明 KKT 条件下有强对偶性

$$\min\max f = \max\min f.$$

在 svm 中 $\lambda = 0$ 是没用的点，$\lambda > 0$ 则说明 $g_i = 0$ 即这个点刚好在最优边界上。

3.3.2 支持向量机优化求解

由强对偶性，可以很容易求解之前的问题，先对 w 求偏导，可以知道问题转化为 $\sum_{i=1}^{m} \lambda_i y^i = 0$ 时求解

$$\max_{\lambda} \left[\sum_{i=1}^{m} \lambda_i + \frac{1}{2} \lambda_i \lambda_j y^{(i)} y^{(j)} \left((x^{(i)})^T x^{(j)} \right) \right].$$

可以使用序列最小优化 (Sequential Minimization Optimization) 来求解求解到 λ 之后可以直接求解 w, b.

线性不可分情况:
应用场景中，数据往往**线性不可分**。在 SVM 中我们一般加入松弛变量 ξ (但需要注意的是引入松弛变量只能解决噪声问题，如果真的不可分则需采用核方法)

$$y^{(i)} \left(w^T x^{(i)} + b \right) \geq 1 - \xi_i, \quad \xi_1 \geq 0.$$

而拉格朗日函数也会相应的改变。我们要求 $\lambda_i \leq C$. **损失函数对比**:
我们称 SVM 的损失函数为 Hinge Loss:

$$\frac{1}{2}\|w\|^2 + C \sum_{i=1}^{m} \max\left(0, 1 - y^{(i)} \left(w^T x^{(i)} + b \right)\right).$$

它与 logistic loss 的区别在于只要支持向量大于一了，就没有相关的 loss 了。

3.3.3 SMO 算法

SMO 的思想也叫坐标上升法（Coordinate Ascent），与梯度下降的区别在于 SMO 每次优化会固定部分坐标，只优化一个维度。考虑到 λ_i 之间是互相有约束的

$$\sum_{i=1}^{m} \lambda_i y^i = 0.$$

我们选取两个变量 λ_i, λ_j 进行更新，对 $W(\lambda)$ 进行优化。例如 $i = 1, j = 2$。其他变量为常数，我们的约束变为

$$\lambda_1 y^{(i)} + \lambda_2 y^{(2)} = -\sum_{i=3}^{m} \lambda_i y^{(i)} = \zeta.$$

最后的优化变成一个二次函数

$$W(\lambda_2) = a\lambda_2^2 + b\lambda + c.$$

第四章　Mar 23

4.1　支持向量机核方法

之前提到过，对 x 进行一个 ϕ 映射可以处理一些非线性问题。而

$$K\left(x^{(i)}, x^{(j)}\right) = \phi\left(x^{(i)}\right)^T \phi\left(x^{(j)}\right).$$

我们期望 $x^{(i)}, x^{(j)}$ 越接近，对应 K 函数越大。**Kernel Trick** 是说我们甚至可以不管 ϕ 直接定义 K 函数。

核函数实际上是一个 similarity measure, 最常用的核函数是高斯核函数

$$K(x, z) = \exp\left(-\frac{\|x - z\|^2}{2\sigma^2}\right).$$

也称为径向基函数 (RBF) 核，ϕ 很复杂并且不重要，所以无需写出。

核矩阵有着很好的性质：

Theorem 4.1.1 核矩阵是对称矩阵和半正定矩阵。

有效核

Definition 4.1.1 给定 $K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, 如果任意 $\{x_1, \cdots, x_n\}$ 对应的核矩阵是半正定矩阵，那么这个核是有效核（*Valid Kernel*[a]）。

[a]也叫 Mercy Kernel

大部分常用核都是有效核。还有一种特殊的核叫做 **sigmoid 核**：

$$K(x, z) = \tanh\left(\alpha x^T z + c\right).$$

可以将它看成一个两层的神经网络。

支持向量机可以看成一个广义的线性模型，基于统计的机器学习的本质思维方式是看两个数据的相似度!

4.2 人工神经网络

4.2.1 感知机模型

Rosenblatt 在 1958 年提出单层感知机的监督学习。考虑预测

$$\hat{y} = \phi\left(\sum_{i=1}^{m} w_i x_i + b\right).$$

其中 ϕ 为激活函数。训练时使用如下方法更新

$$w_i = w_i + \eta\left(y - \hat{y}\right) x_i, \quad b = b + \eta\left(y - \hat{y}\right).$$

这其实就是现在梯度下降的雏形。Rosenblatt 还证明了这个算法在线性可分数据上的收敛性。但是 1969 年时，Minsky 又证明了这种机器学习的方法甚至不能解决一些基本的问题，比如异或。由于缺少有效的算法，人们离开神经网络范式 20 年。

4.2.2 隐藏层和反向传播

到了 1986 年，人们提出添加隐藏层去学习线性不可分的数据。但是隐藏层的添加方法并不唯一，于是又有了 **feed foward** 的神经网络

Definition 4.2.1 前馈神经网络，即消息从输入节点出发，经过隐藏节点传到输出节点，且网络中没有或者循环。

多层的神经网络一定会有非线性的激活函数，常用的激活函数有

- Sigmoid 函数
$$\sigma(z) = \frac{1}{1 + e^{-z}}.$$
- tanh
$$\tanh(x) = \frac{1 - e^{2z}}{1 + e^{2z}}.$$
- Rectified linear unit
$$ReLU(z) = \max(0, z).$$

4.2.3 普适逼近定理

下面这个定理揭示了为什么神经网络能过如此有效

Theorem 4.2.1 一个具有至少一层隐藏层的前馈神经网络，并且隐藏层包含有限数量的神经元，他可以以任意精度逼近定义在 $\mathbb{R}^n$ 的闭集里的连续函数。(前提是这个神经网络的激活函数满足某些性质，如 *relu,sigmoid*)

4.2.4 反向传播算法

反向传播算法于 1986 年提出，而现在还在广泛使用。反向传播算法的本质就是函数求导的链式法则。具体推导见wiki。