机器学习与数据挖掘

1 机器学习介绍

1.1 机器学习用于?

给一个输入x，预测一个输出y。

机器学习帮助找到最好的预测函数。

1.2 什么是机器学习

不同于基于规则的系统。

机器学习算法：根据输入的训练数据（一组（x，y）对），输出一个预测函数（输入x预测输出y）（监督学习）。

常见机器学习问题类型：分类、回归。

1.3 特征提取

定义：把原始输入x映射到R^d。

更好的特征使得所需要的机器学习方法可以更简单。

独热编码：二进制特征集中，只有一个非零值。

1.4 预测函数的评估

损失函数：反应预测值和目标输出的差距。

分类误差、平方误差。

训练集用于训练预测函数， 验证集用于调参/选择最优的预测函数，测试集用于评估性能。

K折交叉验证，当测试集对于好的性能评估过少时使用。

1.5 泄露 偏差 不稳定性

泄露：关于标签的信息泄露到特征中。

偏差：测试输入和部署输入分布不同。

不稳定性：协变量偏移：训练和部署的输入分布发生变化；概念漂移：正确的输出随时间改变。

1.6 模型复杂度，过拟合

更大的模型复杂度：更拟合训练数据，但是不一定在测试数据上有更好的性能。

防止过拟合：减少模型复杂度，使用更多的训练数据。

超参数：模型复杂度、模型复杂度类型（不同正则项）、优化算法（学习率）、模型类型（损失函数、核函数类型）。

2 统计学习理论

2.1 动作

动作是模型产生的东西的泛称。评价标准基于找到最优的动作。

outcome（真实结果）和action通常是独立的。

预测函数：输入input x，输出预测a。

损失函数：输入预测a和真实结果y，输出损失值。损失函数评估单个动作。

2.2 统计学习理论

独立同分布：没有协变量偏移和数据漂移。

假设数据生成分布为$P_{x\times y}$，则预测函数的**风险**为

$$R(f)=E(l(f(x),y))$$

即从分布$P_{x\times y}$中随机取出的样本的**期望损失**。由于分布是未知的，所以不能直接计算出期望损失，而是通过**贝叶斯风险**估计。

贝叶斯决策函数是在所有可选函数中，选择最小风险的函数的函数，也称为目标函数。

$$f^*=\arg\min_f R(f)$$

贝叶斯决策函数的风险称为**贝叶斯风险**。

令$D_n=((x_1,y_1),\dots,(x_n,y_n))$从分布$P_{x\times y}$中取出，则$f$对于$D_n$的**经验风险**为

$$\hat{R}_n(f)=\frac{1}{n}\sum_{i=1}^{n}l(f(x_i),y_i)$$

则有

$$\lim_{n\to\infty}\hat{R}_n(f)=R(f)$$

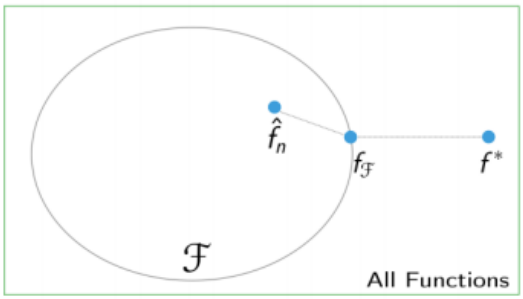
经验风险最小化函数定义为

$$\hat{f}=\arg\min_f \hat{R}_n(f)$$

如何从训练输入泛化到其他新输入？比起对所有决策函数中选择经验风险最小的函数，不如把可选函数（易于计算的、平滑的）限制在一个子集中，称为**假设空间**。假设空间中的风险最小化函数记为f_F，经验风险最小化函数记为$\hat{f}_n$。

2.3 代价分解

误差分解：



$$f^* = \arg\min_f \mathbb{E}\ell(f(X), Y)$$
$$f_{\mathcal{F}} = \arg\min_{f\in\mathcal{F}} \mathbb{E}\ell(f(X), Y))$$
$$\hat{f}_n = \arg\min_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i), y_i)$$

- **Approximation error** (of $\mathcal{F}$) $= R(f_{\mathcal{F}}) - R(f^*)$
- **Estimation error** (of $\hat{f}_n$ in $\mathcal{F}$) $= R(\hat{f}_n) - R(f_{\mathcal{F}})$

近似误差：假设空间的选择是否合适。

估计误差：反应用训练数据所得到的f能达到的性能程度。

当扩大假设空间，近似误差减小，估计误差通常变大（因为搜索范围增大导致过拟合风险增加）。

超额风险excess risk比较一个函数和贝叶斯最优函数。超额风险不可能为负数。

$$ExcessRisk(f) = R(f) - R(f^*)$$

- The excess risk of the ERM $\hat{f}_n$ can be decomposed:

$$\begin{aligned}\textbf{Excess Risk}(\hat{f}_n) &= R(\hat{f}_n) - R(f^*) \\ &= \underbrace{R(\hat{f}_n) - R(f_{\mathcal{F}})}_{\text{estimation error}} + \underbrace{R(f_{\mathcal{F}}) - R(f^*)}_{\text{approximation error}}.\end{aligned}$$

优化误差：设$\tilde{f}_n \in F$是最优化方法返回的函数，$\hat{f}_n$是经验风险最小化函数，则优化误差为：

$$OptimizationError = R(\tilde{f}_n) - R(\hat{f}_n)$$

优化误差可能为负（过拟合），但是若取经验风险而非风险，则差值大于等于零。

$$\begin{aligned}\textbf{Excess Risk}(\tilde{f}_n) &= R(\tilde{f}_n) - R(f^*) \\ &= \underbrace{R(\tilde{f}_n) - R(\hat{f}_n)}_{\text{optimization error}} + \underbrace{R(\hat{f}_n) - R(f_{\mathcal{F}})}_{\text{estimation error}} + \underbrace{R(f_{\mathcal{F}}) - R(f^*)}_{\text{approximation error}}\end{aligned}$$

3 梯度以及随机梯度下降

3.1 梯度下降

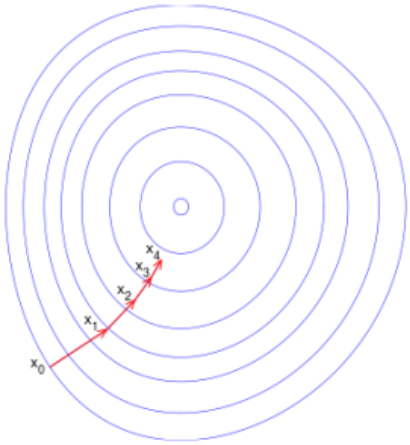
梯度下降是一种求解可微函数局部最小值的一阶迭代优化算法。

Gradient Descent

- Initialize $x = 0$
- Repeat

$$x \leftarrow x - \underbrace{\eta}_{\text{step size}} \nabla f(x)$$

- until stopping criterion satisfied



步长：若函数满足李普希兹连续，则当固定步长小于等于1/L时收敛。

何时停止：当梯度小于一个自己设置的ϵ时停止。对于模型学习，每次用验证集评估性能，当性能不再提升时停止。

3.2 经验风险的梯度下降

- Suppose we have a hypothesis space of functions $\mathcal{F} = \{ f_w : \mathcal{X} \to \mathcal{A} \mid w \in \mathbf{R}^d \}$
  - Parameterized by $w \in \mathbf{R}^d$.
- ERM is to find w minimizing

$$\hat{R}_n(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(f_w(x_i), y_i),$$

- Suppose $\ell(f_w(x_i), y_i)$ is differentiable as a function of w.
- Then we can do gradient descent on $\hat{R}_n(w)$ ...

- At every iteration, we compute the gradient at current w:

$$\nabla \hat{R}_n(w) = \frac{1}{n} \sum_{i=1}^{n} \nabla_w \ell(f_w(x_i), y_i)$$

- We have to touch all n training points to take a single step. $[O(n)]$

为了减少计算时间，采用梯度的估计：minibatch梯度，即选择部分样本求梯度均值作为全体样本梯度均值的估计值。

minibatch的batchsize越大，则对梯度的估计越好，但是越慢。若batchsize=1，则称为随机梯度下降法（SGD）。

自适应步长算法。

4 L1 L2正则化

正则化：避免过拟合的一种思路，通过引入惩罚项来压制参数大小。

4.1 岭回归

岭回归：经验误差+L2正则项（正则化参数$\lambda \geq 0$ * 参数二范数的平方）：

$$\hat{w} = \arg\min_{w \in R^d} \frac{1}{n} \sum_{i=1}^{n} \{w^T x_i - y_i\}^2 + \lambda ||w||_2^2$$

4.2 Lasso回归

Lasso回归：经验误差+L1正则项（正则化参数$\lambda \geq 0$ * 参数一范数）

$$\hat{w} = \arg\min_{w \in R^d} \frac{1}{n} \sum_{i=1}^{n} \{w^T x_i - y_i\}^2 + \lambda ||w||_1$$

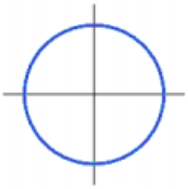
Lasso倾向于将参数变为0，而岭回归则是收缩参数，但通常不会将他们收缩到0。因此，Lasso可以选出重要的参数（非0的），得到稀疏解，以节省开销。

4.3 为什么Lasso回归会得到稀疏解

- For visualization, restrict to 2-dimensional input space
- $\mathcal{F} = \{f(x) = w_1 x_1 + w_2 x_2\}$ (linear hypothesis space)
- Represent $\mathcal{F}$ by $\{(w_1, w_2) \in \mathbf{R}^2\}$.

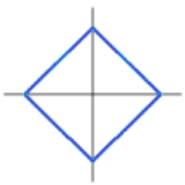
L_2 contour:

$w_1^2 + w_2^2 = r$

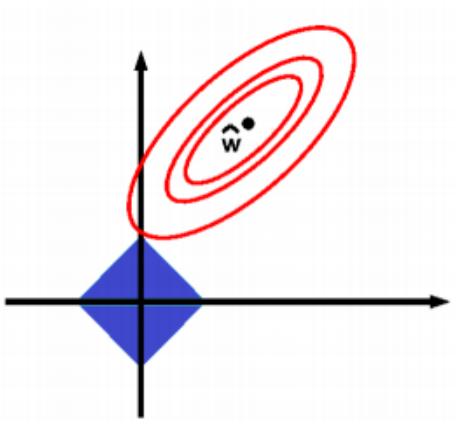


L_1 contour:

$|w_1| + |w_2| = r$



- $f_r^* = \arg\min_{w \in \mathbf{R}^2} \frac{1}{n} \sum_{i=1}^{n} (w^T x_i - y_i)^2$ subject to $|w_1| + |w_2| \le r$



- Blue region: Area satisfying complexity constraint: $|w_1| + |w_2| \le r$
- Red lines: contours of $\hat{R}_n(w) = \sum_{i=1}^{n} (w^T x_i - y_i)^2$

4.4 坐标下降法

用以找到Lasso解。

目标：最小化$L(w) = L(w_1, \dots w_d)$。

对于梯度下降和随机梯度下降，每一步都会改变所有w_i，而在坐标下降法中，一步只改变一个w_i。

Coordinate Descent Method

Goal: Minimize $L(w) = L(w_1, \dots w_d)$ over $w = (w_1, \dots, w_d) \in \mathbf{R}^d$.

- **Initialize** $w^{(0)} = 0$
- **while** not converged:
  - Choose a coordinate $j \in \{1, \dots, d\}$
  - $w_j^{\text{new}} \leftarrow \arg\min_{w_j} L(w_1^{(t)}, \dots, w_{j-1}^{(t)}, \mathbf{w_j}, w_{j+1}^{(t)}, \dots, w_d^{(t)})$
  - $w_j^{(t+1)} \leftarrow w_j^{\text{new}}$ and $w^{(t+1)} \leftarrow w^{(t)}$
  - $t \leftarrow t + 1$
- Random coordinate choice $\Rightarrow$ **stochastic coordinate descent**
- Cyclic coordinate choice $\Rightarrow$ **cyclic coordinate descent**

对于Lasso，坐标下降法有解析解。

坐标下降法收敛到全局最优的条件：

Theorem

[a]If the objective f has the following structure

$$f(w_1, \ldots, w_d) = g(w_1, \ldots, w_d) + \sum_{j=1}^{d} h_j(w_j),$$

where

- $g : \mathbf{R}^d \to \mathbf{R}$ *is differentiable and convex, and*
- *each* $h_j : \mathbf{R} \to \mathbf{R}$ *is convex (but not necessarily differentiable)*

then the coordinate descent algorithm converges to the global minimum.

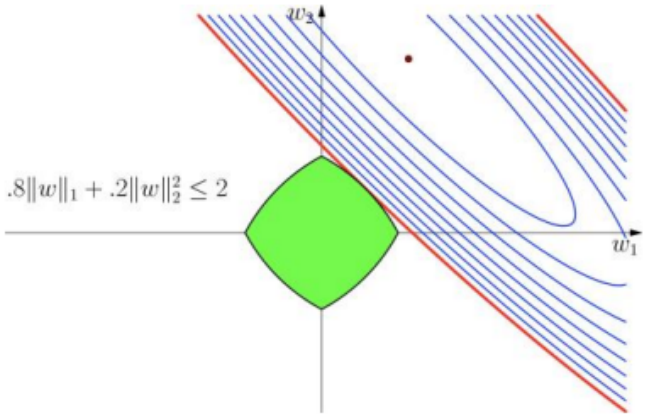
5 弹性网络

对于存在冗余特征的情况，会有多个经验风险最小化函数ERM。若导入L1和L2的正则项，则对于L1，当参数符号相同时不存在歧视，而对于L，当所有特征平等扩展时，风险最小。

当多个参数相关时，保留这些参数，并给出相同的权重，可以抵消误差，因此需要用到弹性网络。使用Lasso时，对于参数相关的情况，则倾向于随机保留其中一个。

弹性网络将Lasso和岭回归合并：

$$\hat{w} = \arg\min_{w \in R^d} \frac{1}{n} \sum_{i=1}^{n} \{w^T x_i - y_i\}^2 + \lambda_1 ||w||_1 + \lambda_2 ||w||_2^2$$



- Elastic net solution is closer to $w_2 = w_1$ line, despite high correlation.

6 凸优化

把问题转化为凸优化问题，就算没有解析解，也可以有高效的算法找到最优解。

对偶问题：引入拉格朗日乘子。

The general [inequality-constrained] optimization problem is:

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \le 0, \ \ i = 1, \ldots, m \end{array}$$

Definition

The **Lagrangian** for this optimization problem is

$$L(x, \lambda) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x).$$

- We refer to λ_i as **Lagrange multipliers** (also called the **dual variables**).

Definition

We define the **Lagrange dual function** $g : \mathbf{R}^m \to \mathbf{R}$ as the minimum value of the Lagrangian over x: for $\lambda \in \mathbf{R}^m$,

$$g(\lambda) = \inf_x L(x, \lambda) = \inf_x \left(f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) \right).$$

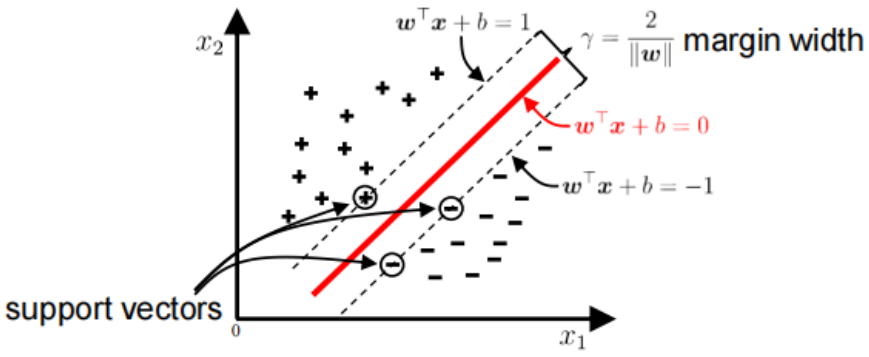
- The dual function may take on the value $-\infty$ (e.g. $f_0(x) = x$).

7 SVM

7.1 间隔

间隔：选择最中间的，则具有最大的容差、高鲁棒性、强泛化能力。

Hyperplane: $\boldsymbol{w}^\top \boldsymbol{x} + b = 0$



7.2 SVM【公式都默一下，见西瓜书P124】

SVM基本形式：

- Maximizing the margin:

$$\begin{aligned} &\arg\max_{\boldsymbol{w},b} \ \frac{2}{\|\boldsymbol{w}\|} \\ &\text{s.t.} \ \ y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1, \ \ i = 1, 2, \ldots, m. \end{aligned}$$

$$\begin{aligned} &\arg\min_{\boldsymbol{w},b} \ \frac{1}{2}\|\boldsymbol{w}\|^2 \\ &\text{s.t.} \ \ y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1, \ \ i = 1, 2, \ldots, m. \end{aligned}$$

是一个二次规划问题，其对偶问题形式：

- Step 1: Lagrangian

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{m} \alpha_i \left(y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) - 1\right)$$

- Step 2: Calculate the derivatives of $L(\boldsymbol{w}, b, \boldsymbol{\alpha})$ w.r.t $\boldsymbol{w}$ and b

$$\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i, \quad \sum_{i=1}^{m} \alpha_i y_i = 0.$$

- Step 3: Replace *w* back:

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \ \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x}_j - \sum_{i=1}^{m} \alpha_i \\ &\text{s.t.} \ \ \sum_{i=1}^{m} \alpha_i y_i = 0, \ \ \alpha_i \geq 0, \ \ i = 1, 2, \ldots, m. \end{aligned}$$

由KKT条件得

$$\begin{cases} \alpha_i \geq 0, \\ y_i f(\boldsymbol{x}_i) \geq 1, \\ \alpha_i (y_i f(\boldsymbol{x}_i) - 1) = 0. \end{cases}$$

因此最终模型只和支持向量有关。

7.3 确认参数

α：采用SMP算法

每次选择两个变量α_i和α_j，并固定其他参数，代入到约束中使得α_j可用α_i表示，求解对偶问题以更新这两个变量得值。

b：对于每一个支持向量，计算对应的b求均值。

7.4 线性不可分情况：核函数、软间隔

核函数

Kernelized SVM

Dual form:

$$\min_{\boldsymbol{\alpha}} \ \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \boxed{\phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}_j)} - \sum_{i=1}^{m}\alpha_i$$
$$\text{s.t.} \ \sum_{i=1}^{m}\alpha_i y_i = 0, \ \alpha_i \geq 0, \ i = 1, 2, \ldots, m.$$

Inner production form

Prediction function:

$$f(\boldsymbol{x}) = \boldsymbol{w}^\top \phi(\boldsymbol{x}) + b = \sum_{i=1}^{m}\alpha_i y_i \boxed{\phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x})} + b$$

用核函数表示内积，使得我们可以在原始特征空间中操作，而不需要计算高维空间中的数据坐标。

Basic idea: Instead of explicitly designing mapping function, we design kernel functions:

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}_j)$$

Common kernel functions:

名称	表达式	参数
线性核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^\top \boldsymbol{x}_j$	
多项式核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^\top \boldsymbol{x}_j)^d$	$d \geq 1$为多项式的次数
高斯核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\delta^2}\right)$	$\delta > 0$为高斯核的带宽(width)
拉普拉斯核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}{\delta}\right)$	$\delta > 0$
Sigmoid核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh(\beta \boldsymbol{x}_i^\top \boldsymbol{x}_j + \theta)$	tanh为双曲正切函数, $\beta > 0, \theta < 0$

软间隔

软间隔允许一些样本不满足SVM的限制（分错类）。

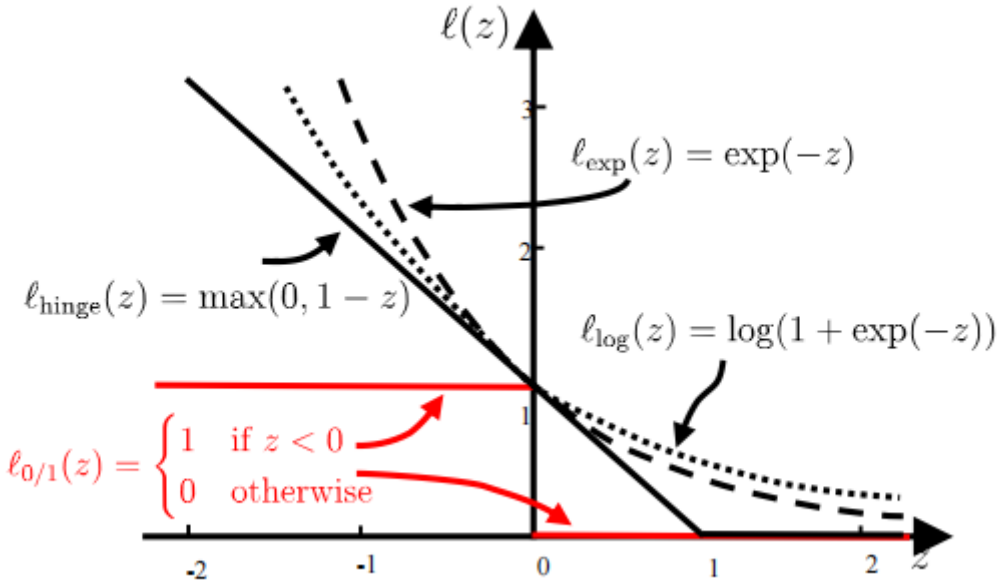
思想：最大化间隔，最小化不满足限制的样本。

$$\min_{\boldsymbol{w}, b} \ \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m} l_{0/1}\left(y_i(\boldsymbol{w}^\top \phi(\boldsymbol{x}_i) + b) - 1\right)$$

$l_{0/1}$ is 0/1 loss function: $l_{0/1} = \begin{cases} 1 & z < 0 \\ 0 & otherwise \end{cases}$

Problem: 0/1 loss function is nonconvex and discontinuous, difficult to optimize.

Hinge Loss



当分类正确时，损失为0，否则离分类正确越远，损失越大。

Original objective function:

$$\min_{\boldsymbol{w},b} \ \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m} \max\left(0, 1 - y_i(\boldsymbol{w}^\top \phi(\boldsymbol{x}_i) + b)\right)$$

Dual form:

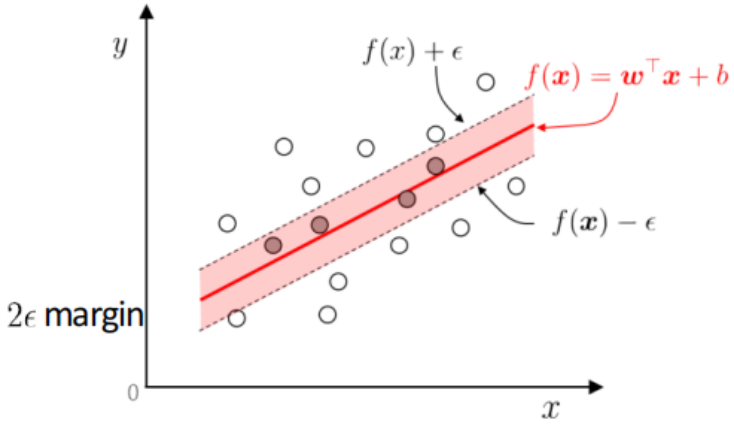
$$\min_{\boldsymbol{\alpha}} \ \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}_j) - \sum_{i=1}^{m} \alpha_i$$

$$\text{s.t.} \ \sum_{i=1}^{m} \alpha_i y_i = 0, \ 0 \le \alpha_i \le C, \ i = 1, 2, \ldots, m.$$

7.5 SVM回归

对于岭回归，可以引入核函数变成核岭回归。由于对于岭回归，结果和所有样本有关，使得计算时间慢，因此可以通过改变损失函数来让问题变稀疏。

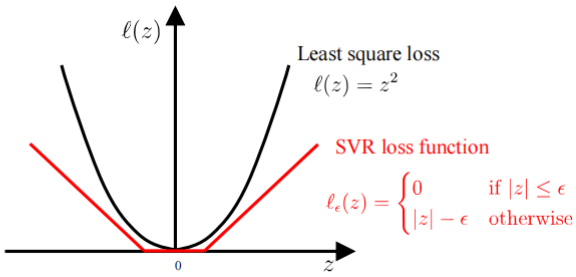
Feature: allows to have a 2ϵ deviation between model output and actual output.



The SVR problem can be formalized as follows:

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}\ell_\epsilon(f(\boldsymbol{x}_i) - y_i)$$

$$\ell_\epsilon(z) = \begin{cases} 0 & \text{if } |z| \leq \epsilon \\ |z| - \epsilon & \text{otherwise} \end{cases}$$



由于在边界内的不算损失，因此模型变稀疏。

形式化定义：

$$\begin{aligned} \min_{\boldsymbol{w},b,\xi_i,\hat{\xi}_i} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}(\xi_i + \hat{\xi}_i) \\ \text{s.t.} \quad & y_i - \boldsymbol{w}^\top\phi(\boldsymbol{x}_i) - b \leq \epsilon + \xi_i, \\ & y_i - \boldsymbol{w}^\top\phi(\boldsymbol{x}_i) - b \geq -\epsilon - \hat{\xi}_i, \\ & \xi_i \geq 0,\ \hat{\xi}_i \geq 0,\ i = 1, 2, \ldots, m. \end{aligned}$$

7.6 SVM多分类

一对多OVR：自己一类为正集 其余类为负集，N个类别（N个label）有N个训练集。

优点：训练分类器个数较少，分类速度相对较快。

缺点：训练速度随样本数量增加而极具减慢。负类样本远大于正类样本，导致样本不对称。一个样本可能同时属于几个类或不属于任何一个。

一对一OVO：任意两个样本之间设计一个SVM，最后通过投票分数来看分类情况。

优点：不会出现样本不属于任何一类的情况。

缺点：复杂度大。

7.7 课堂测试

hinge loss？控制多少是允许分错类的。

SVM回归？允许有间隔在的回归模型。

SVM中的C？软间隔中的惩罚项权重。C越小，给约束的权重越小，允许更多的错误分类。C越大，更注重正确分类，容易过拟合。

SVM是否会被异常点影响？无间隔情况下，支持向量是异常点，则会。软间隔情况下，受异常点影响取决于C的大小。

可以把核函数用在逻辑回归上吗？因为逻辑回归要考虑所有点，所以计算效率低下。

逻辑回归和SVM区别？逻辑回归由所有点决定。

SVM能得到概率输出吗？不能，只能得到二元结果。

8 性能评估

8.1 零信息预测函数

对于分类问题，输出出现频率最高的分类。

对于回归问题，平方损失的零信息预测函数为标签的均值，绝对误差的则是标签的中位数。

8.2 正则化线性模型

通常用正则化线性模型、SVM作为基线模型。

若自己的模型打不过极限模型，则说明自己模型有错或者训练数据过少。

8.3 描述分类性能

混淆矩阵

		Actual	
		Class 0	Class 1
Predicted	Class 0	*a*	*b*
	Class 1	*c*	*d*

正确率：

$$\frac{a+d}{a+b+c+d}$$

错误率：

$$\frac{b+c}{a+b+c+d}$$

- Let's denote the **positive** class by + and **negative** class by −:

		Actual	
		Class +	Class −
Predicted	Class +	TP	FP
	Class −	FN	TN

查准率：

$$\frac{TP}{TP+FP}$$

召回率：

$$\frac{TP}{TP+FN}$$

F1-score：

$$2\frac{presicion*recall}{presicion+recall}$$

Fβ-score：如果需要对准确率和召回率进行加权：

$$(1+\beta^2)\frac{presicion*recall}{\beta^2*presicion+recall}$$

- **True positive rate** is the accuracy on the positive class: TP / (FN + TP)
  - same as **recall**, also called **sensitivity**
- **False negative rate** is the error rate on the positive class: FN / (FN + TP) (“**miss rate**”)
- **False positive rate** is error rate on the negative class: FP / (FP + TN)
  - also called **fall-out** or **false alarm rate**
- **True negative rate** is accuracy on the negative class: TN / (FP + TN) (“**specificity**”)

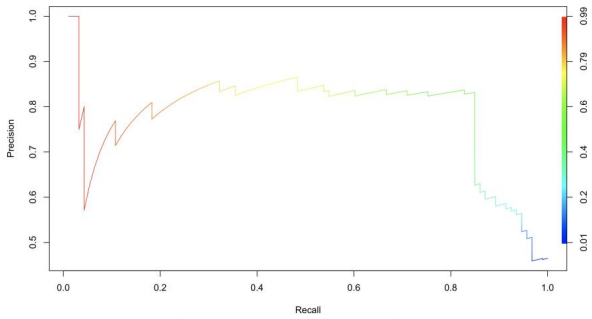
8.4 阈值分类评分函数

假设预测函数为$f:X\to R$，则需要决定$f(x)$在什么范围时预测一个类，反之预测另一个类，即确定阈值。因此需要选择阈值使得评分最大化。

8.5 性能曲线

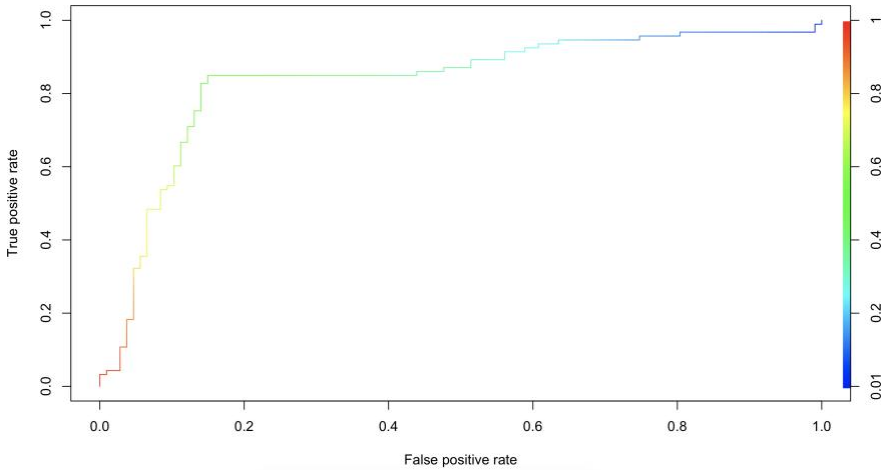
查准率-召回率曲线

当阈值从正无穷降到负无穷时，召回率上升，而准确率变化不定。



ROC曲线

考虑FPR和TPR，它们都是一种召回率，所以当阈值从正无穷降到负无穷时，FP和TP都增加。



使用AUC ROC：即area under the ROC curve来衡量分类模型性能。

9 最大似然估计

9.1 估计一个概率分布

假设这个分布为$p(y)$，而我们估计得到$\hat{p}(y)$，用此来描述未来的数据。

假设有从真实的分布$p(y)$得到的数据集$D = (y_1, \ldots, y_n)$，则$\hat{p}$关于数据集D的似然likelihood定义为

$$\hat{p}(D) = \prod_{i=1}^{n} \hat{p}(y_i)$$

9.2 最大似然估计

假设参数模型为$p(y;\theta),\theta \in \Theta$,则对于特定$\hat{\theta}$的似然为

$$\hat{p}(D;\hat{\theta}) = \prod_{i=1}^{n} \hat{p}(y_i;\hat{\theta})$$

可以转换为log-likelihood，用累加代替累成：

$$\log \hat{p}(D;\hat{\theta}) = \sum_{i=1}^{n} \log \hat{p}(y_i;\hat{\theta})$$

对于模型$p(y;\theta),\theta \in \Theta$的**最大似然估计MLE**：

$$\hat{\theta} = \arg\max_{\theta \in \Theta} \sum_{i=1}^{n} \log \hat{p}(y_i;\theta)$$

找到MLE是一个最优化问题，对于一些模型，MLE的计算有解析解，也可以通过数学方法来计算MLE，比如随机梯度下降。

MLE可能存在过拟合。

对于多个模型的选择，可以选择其中在验证集上最高似然的一个。

10 决策树

对于结构化数据，手工方法更好，对于非结构化数据，构建神经网络更好。

10.1 决策树结构

每个内部节点对应一个测试。

每个分支对应一个测试结果。

每个叶子节点赋予一个分类。

10.2 熵

衡量不确定性。

$$H[x] = -\sum_{x} p(x) \log_2 p(x)$$

信息增益

$$Gain(D,a) = Ent(D) - \sum_{v=1}^{V} \frac{|D^v|}{|D|} Ent(D^v)$$

信息增益越大，通过属性a划分得到的纯度提升越大，越好。

对应于ID3算法。

增益率Gain Ratio

$$Gain\_ratio(D,a)\frac{Gain(D,a)}{IV(a)}$$

$$IV(a)=-\sum_{v=1}^{V}\frac{|D^v|}{|D|}\log_2\frac{|D^v|}{|D|}$$

划分的数量（即属性a的取值）越多，增益率越小。

增益率越大越好。

对应于C4.5算法。

Gini系数

不使用熵，而是使用不纯度来进行衡量。Gini越小越好。

$$Gini(D)=1-\sum_{k=1}p_k^2$$

$$Gini\_index(D,a)=Gini(D)-\sum_{v=1}^{V}\frac{|D^v|}{|D|}Gini(D^v)$$

Gini_index越大越好。

对应于CART算法。

10.3 属性离散化

对于连续值的属性，进行离散化。

把一个属性的可能取值划分为多个区间，取中值作为这个区间的取值。每次计算特定属性的信息增益时，在所有的取值中选择最好的。

10.4 处理缺失属性值

简单方法：使用该节点的所有样本中出现最多的值，或者该节点中多数的类别的样本中，出现最多的值。

10.5 过拟合

决策树容易过拟合，为了防止过拟合，有两种方法：

提前结束：在树完美分类所有训练数据之前结束。

剪枝：先得到完整的树，再剪枝（用一个叶子节点来代替子树，对应的分类是子树对应样本中最多数的分类）。

预剪枝：根据树的深度限制、每个结点最少样本数的限制。

后剪枝：衡量剪枝带来的精度影响再决定要不要剪枝。

剪枝的衡量

通过验证集或者训练集（看扩大或修建能够否对训练集结果产生改进）。

Reduce-Error Pruning：

在进一步剪枝有害之前，衡量每一个可能剪枝节点对验证集结果带来的影响，贪心地移走对验证集准确率提升最大的子树。

11 集成学习1

集成模型能够集成3多个模型，以获得比单个模型更好的结果。这些算法可以是不同或相同的。

并行集成：每个模型独立构建，如bagging，随机森林。主要思想是结合高复杂性，低偏差的模型来减少方差。

顺序集成：添加能弥补之前模型缺陷的新模型在后面。

11.1 Bagging

基本思想

基于bootstrap sampling，进行有放回的采样，直到获得和原数据集大小一样的采样数据集。采样多个大小金额原数据集大小一样的采样集，对于每个采样集训练出基学习器，再将基学习器进行结合（如投票法）。

OOB预测

- For i-th training point, let

$$S_i = \{b \mid D^b \text{ does not contain } i\text{th point}\}.$$

- The **OOB prediction** on x_i is

$$\hat{f}_{\text{OOB}}(x_i) = \frac{1}{|S_i|} \sum_{b \in S_i} \hat{f}_b(x_i).$$

优点

低时间复杂度。每个基学习器之间没有强依赖性。并行生成。

什么时候使用Bagging?

当单个模型不稳定的时候（训练集的轻微改变导致分类器的巨大变化）

Bagging可以综合投票的结果，来提升稳定性和准确性。

不稳定的模型：决策树，神经网络。

11.2 随机森林

随机森林是一个经典的bagging算法。

主要特征：样本有放回采样，属性随机采样，基学习器通常为决策树。

决策树的限制

决策树易进入局部最优，因为用的贪心算法。

当单个决策树决定分类边界时，由于只涉及单个特征的逻辑判断，分类边界是一条平行于坐标轴的直线，这使得模型表达能力受到限制，精度较差。

随机森林使得结果更加准确。

随机森林的随机性

为了保证每个树的独立性，需要保证两个随机性：随机选择数据（bagging），随机选择属性。由于具有这些随机性，因此不需要剪枝。

随机森林算法

Algorithm 1 Random Forest

Precondition: A training set $S := (x_1, y_1), \ldots, (x_n, y_n)$, features F, and number of trees in forest B.

```
function RandomForest(S, F)
    H ← ∅
    for i ∈ 1, ..., B do
        S^(i) ← A bootstrap sample from S
        h_i ← RandomizedTreeLearn(S^(i), F)
        H ← H ∪ {h_i}
    end for
    return H
end function
function RandomizedTreeLearn(S, F)
    At each node:
        f ← very small subset of F
        Split on best feature in f
    return The learned tree
end function
```

每棵树做一个预测，整个森林进行投票，多数票的结果作为输出。

随机森林的衡量

Margins：

单个样本的间隔：正确分类决策树的比例减去错误分类决策树的比例。

森林的间隔：对所有样本的间隔求平均。

OOB Error：

对每个样本，让把它作为OOB样本的树去做预测，并投票。

统计所有样本中分错的比例，作为OOB误差。

优点

可以处理高维数据，而不需要进行特征选择。

可以知道哪些属性更重要。

易于并行化。

缺点

与决策树相比速度更慢。

与决策树相比，可解释性更差。

衡量变量重要性

记录每棵树对于OOB样本的预测正确率。随机变化OOB样本的第 j 列的顺序再记录准确率。这种变化的精度下降对所有树取平均，用于衡量随机森林中变量 j 的重要性。

11.3 课堂练习

随机森林的随机性？样本随机性，属性随机性。

为什么随机森林不容易过拟合？基于随机+多棵树投票。

为什么随机森林不用完整样本训练多个随机树？随机性下降，各数据集间相关性上升，导致过拟合，且没有OOB样本。

12 集成学习2

12.1 Boosting

从初始训练集训练出一个基学习器，再根据其表现，对训练样本分布进行调整，使得先前基学习器做错的训练样本在后续得到更多关注，然后基于调整过的样本分布来训练下一个基学习器，重复至基学习器的数目达到了某个特定值。最终将这些学习器进行加权结合。

Input: Sample distribution $\mathcal{D}$;
Base learning algorithm $\mathfrak{L}$;
Number of learning rounds T.

Process:

1. $\mathcal{D}_1 = \mathcal{D}$. % Initialize distribution
2. **for** $t = 1, \ldots, T$:
3. $h_t = \mathfrak{L}(\mathcal{D}_t)$; % Train a weak learner from distribution $\mathcal{D}_t$
4. $\epsilon_t = P_{\boldsymbol{x} \sim D_t}(h_t(\boldsymbol{x}) \neq f(\boldsymbol{x}))$; % Evaluate the error of h_t
5. $\mathcal{D}_{t+1} = Adjust\_Distribution(\mathcal{D}_t, \epsilon_t)$
6. **end**

Output: $H(\boldsymbol{x}) = Combine\_Outputs(\{h_1(\boldsymbol{x}), \ldots, h_t(\boldsymbol{x})\})$

12.2 AdaBoost

调整的权重和每个分类器的权重都取决于分类器的错误率。

输入： 训练集 $D=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\dots,(\boldsymbol{x}_m,y_m)\}$;
基学习算法 $\mathfrak{L}$;
训练轮数 T.

过程：

$\mathcal{D}_1(\boldsymbol{x})=1/m.$
for $t=1,2,\dots,T$ **do**
  $h_t=\mathfrak{L}(D,\mathcal{D}_t)$;
  $\epsilon_t=P_{\boldsymbol{x}\sim\mathcal{D}_t}(h_t(\boldsymbol{x})\neq f(\boldsymbol{x}))$;
  if $\epsilon_t>0.5$ **then break**
  $\alpha_t=\frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$;
  $$\begin{aligned}\mathcal{D}_{t+1}(\boldsymbol{x}) &= \frac{\mathcal{D}_t(\boldsymbol{x})}{Z_t}\times\begin{cases}\exp(-\alpha_t), & \text{if } h_t(\boldsymbol{x})=f(\boldsymbol{x})\\ \exp(\alpha_t), & \text{if } h_t(\boldsymbol{x})\neq f(\boldsymbol{x})\end{cases}\\ &= \frac{\mathcal{D}_t(\boldsymbol{x})\exp(-\alpha_t f(\boldsymbol{x})h_t(\boldsymbol{x}))}{Z_t}\end{aligned}$$
end for

输出： $H(\boldsymbol{x})=\operatorname{sign}\left(\sum_{t=1}^{T}\alpha_t h_t(\boldsymbol{x})\right)$

12.3 合并策略

简单平均法、加权平均法、投票法、学习法（用另一个学习器来进行结合，如Stacking）。

12.4 多样性

通过数据采样和考虑属性的扰动来提升学习器多样性。

对于数据采样：不稳定：决策树、神经网络；稳定：线性学习器，SVM，朴素贝叶斯，k近邻。

属性扰动：对于原始属性集，随机采样出多个属性子集，来训练不同的基学习器。

输入： 训练集 $D=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\cdots,(\boldsymbol{x}_m,y_m)\}$;
基学习算法 $\mathfrak{L}$;
基学习器数 T;
子空间属性数 d'.

过程：

for $t=1,2,\dots,T$ **do**
  $\mathcal{F}_t=\mathrm{RS}(D,d')$
  $D_t=\mathrm{Map}_{\mathcal{F}_t}(D)$
  $h_t=\mathfrak{L}(D_t)$
end for

输出： $H(\boldsymbol{x})=\underset{y\in\mathcal{Y}}{\arg\max}\sum_{t=1}^{T}\mathbb{I}\left(h_t\left(\mathrm{Map}_{\mathcal{F}_t}(\boldsymbol{x})\right)=y\right)$

12.5 Stacking

stacking先从初始数据集训练出初级学习器，然后生成一个新的数据集（每个初级学习器对每个样本的输出）来训练次级学习器。

Input: Data set $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
First-level learning algorithms $\mathfrak{L}_1, \ldots, \mathfrak{L}_T$;
Second-level learning algorithm $\mathfrak{L}$.

Process:

for $t = 1, \ldots, T$: % Train a first-level learner by applying the
  $h_t = \mathfrak{L}_t(D)$; % first-level learning algorithm $\mathfrak{L}_t$
end
$D' = \emptyset$; % Generate a new data set
for $i = 1, \ldots, m$:
  for $t = 1, \ldots, T$:
    $z_{it} = h_t(\boldsymbol{x}_i)$;
  end
  $D' = D' \cup ((z_{i1}, \ldots, z_{iT}), y_i)$;
end
$h' = \mathfrak{L}(D')$; % Train the second-level learner h' by applying
% the second-level learning algorithm $\mathfrak{L}$ to the
% new data set $\mathcal{D}'$.

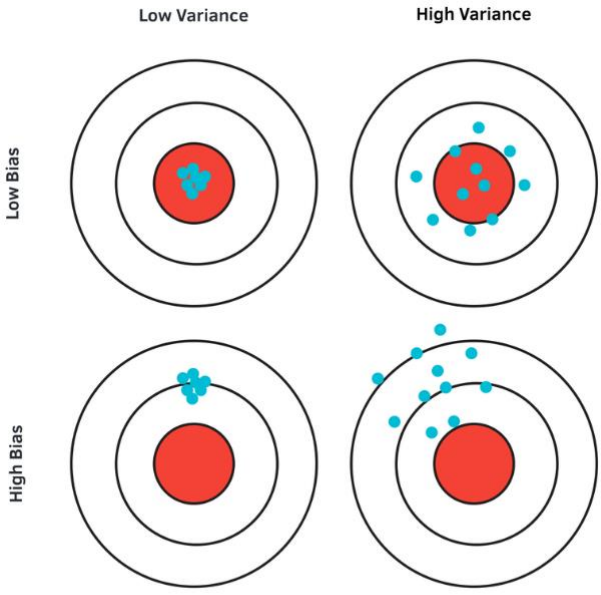
Output: $H(\boldsymbol{x}) = h'(h_1(\boldsymbol{x}), \ldots, h_T(\boldsymbol{x}))$

12.6 课堂测试

解释Bagging减少方差，boosting减少偏差?

bagging对样本重采样，训练不同的模型取平均，因此减少方差。

而boosting每次重点解决前面分错的部分，因此减少偏差。



Bias-Variance Tradeoff

能把随机森林的基分类器换成线性分类器或者k近邻吗？可以。

集成学习只能用在监督学习吗？不一定，如聚类，但是不常用。

集成的分类器不一定比每个基分类器更准确吗？当有的基分类器很弱时，可能集成的还更弱。

深度学习模型和集成模型的关联？多个卷积核等，也体现了集成的思想。

13 PCA主成分分析

维度灾难：稀疏数据样本和距离计算的困难在高维情况下，是机器学习的常见障碍。

因此需要通过降维来避免这些问题。这是一种无监督学习形式，学习内容是从高维可见空间到低维潜在空间的映射。

为什么可以降维

与学习任务密切相关的可能只是一个低维分布，即高维空间的低维嵌入。

虽然降维可能造成数据的丢失，但是增加了样本信息稠密程度，并且一定程度上可以降噪。

线性降维

最简单的方法。用一个转换矩阵进行降维。最常见的线性降维方法是PCA。

13.1 PCA

PCA的基本思想是找到高维数据到低维子空间的正交投影。

希望低维表示是对原始数据的一个好的近似值，因此希望降维再升维回去的数据能和原始数据相近。

$$\mathcal{L}(\mathbf{W}) \triangleq \frac{1}{N}\sum_{n=1}^{N}||\mathbf{x}_n - \text{decode}(\text{encode}(\mathbf{x}_n;\mathbf{W});\mathbf{W})||_2^2$$

转换矩阵是正交矩阵，因为正交矩阵表达能力最优（不线性相关）。

样本需要进行中心化（样本和为0）。

PCA应该找到的投影超平面的性质：

- 最近重构性：样本点到这个超平面的距离足够近。
- 最大可分性：样本点在这个超平面的投影能尽可能分开。

根据这两个性质可以得到最优化问题：

$$\max_W tr(W^T X X^T W)$$
$$s.t. W^T W = I$$

通过拉格朗日乘子法可以得到

$$XX^T w_i = \lambda_i w_i$$

因此只需要对协方差矩阵XX^T进行特征值分解，按照特征值从大到小对应的特征向量组合成转换矩阵即可。

如何选择低维空间维数

通过事先指定。

通过在唯独不同的低维空间用k近邻的交叉验证选取。

设置一个重构阈值 $\frac{\sum_{i=1}^{d'}\lambda_i}{\sum_{i=1}^{d}\lambda_i} \geq t.$ 。

如何处理新样本

先减去原样本的均值（中心化），再乘以转换矩阵。

13.2 核PCA

找到高维到低维的非线性映射。相当于，先往高维映射，再用线性映射到低维。

Kernel PCA (KPCA)

- In general, we don't know the concrete form of ϕ , so we introduce kernel function:

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^{\mathrm{T}} \phi(\boldsymbol{x}_j).$$

- Substituting $\mathbf{W} = \sum_{i=1}^{m} \phi(\boldsymbol{x}_i)\boldsymbol{\alpha}_i$ into $\left(\sum_{i=1}^{m} \phi(\boldsymbol{x}_i)\phi(\boldsymbol{x}_i)^{\mathrm{T}}\right)\mathbf{W} = \lambda\mathbf{W}$:

$$\mathbf{K}\mathbf{A} = \lambda\mathbf{A}.$$

where $\mathbf{K}$ is the corresponding kernel matrix of κ :

$$(\mathbf{K})_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j), \mathbf{A} = (\boldsymbol{\alpha}_1; \boldsymbol{\alpha}_2; \ldots; \boldsymbol{\alpha}_m).$$

因此转换矩阵对K求特征向量即可。

对于新样本，计算方式为：

- For new sample $\boldsymbol{x}$, the j $(j = 1, 2, \ldots, d')$-th coordinate after projection is:

$$\begin{aligned} z_j &= \boldsymbol{w}_j^{\mathrm{T}} \phi(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i^j \phi(\boldsymbol{x}_i)^{\mathrm{T}} \phi(\boldsymbol{x}) \\ &= \sum_{i=1}^{m} \alpha_i^j \kappa(\boldsymbol{x}_i, \boldsymbol{x}). \end{aligned}$$

13.3 流形学习

基于拓扑流形概念。

局部线性嵌入LLE

试图保持邻域内样本之间的线性关系。

LLE先为每个样本找到近邻下标集合，然后基于集合中的样本点对该样本计算线性重构的系数。

输入: 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
近邻参数 k;
低维空间维数 d'.

过程:

1: **for** $i=1,2,\ldots,m$ **do**
2: 确定 $\boldsymbol{x}_i$ 的 k 近邻;
3: 从式(10.27)求得 w_{ij}, $j\in Q_i$;
4: 对于 $j\notin Q_i$, 令 $w_{ij}=0$;
5: **end for**
6: 从式(10.30)得到 $\mathbf{M}$;
7: 对 $\mathbf{M}$ 进行特征值分解;
8: **return** $\mathbf{M}$ 的最小 d' 个特征值对应的特征向量

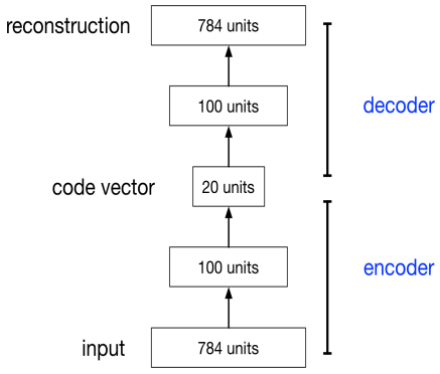
输出: 样本集 D 在低维空间的投影 $Z=\{\boldsymbol{z}_1,\boldsymbol{z}_2,\ldots,\boldsymbol{z}_m\}$.

13.4 自动编码器

自动编码器是一个前馈神经网络。

线性、非线性。

中间：bottleneck layer。



14 聚类1

聚类是常见的无监督学习形式。

14.1 聚类评估指标

外部指标：参考答案，没有标签，只知道哪些应该在一类，因此还是无监督的。

$$a=|SS|,SS=\{(x_i,x_j)|\lambda_i=\lambda_j,\lambda_i^*=\lambda_j^*,i<j\}$$

$$b=|SD|,SD=\{(x_i,x_j)|\lambda_i=\lambda_j,\lambda_i^*\neq\lambda_j^*,i<j\}$$

$$c=|DS|,DS=\{(x_i,x_j)|\lambda_i\neq\lambda_j,\lambda_i^*=\lambda_j^*,i<j\}$$

$$d=|DD|,DD=\{(x_i,x_j)|\lambda_i\neq\lambda_j,\lambda_i^*\neq\lambda_j^*,i<j\}$$

- Jaccard Coefficient (JC)

$$JC = \frac{a}{a+b+c}$$

- Fowlkes and Mallows Index (FMI)

$$FMI = \sqrt{\frac{a}{a+b} \cdot \frac{a}{a+c}}$$

- Rand Index (RI)

$$RI = \frac{2(a+d)}{m(m-1)}$$

内部指标：没有参考答案。类内相似度应该高，类间相似度（最近样本距离）应该低。

- Consider clustering results $C = \{C_1, C_2, \ldots, C_k\}$, define:
  - Average distance between samples in cluster C_l

$$avg(C_l) = \frac{2}{|C_l|(|C_l|-1)} \sum_{1 \le i \le j \le |C_l|} dist(x_i, x_j)$$

  - The longest distance between samples in cluster C_l

$$diam(C_l) = max_{1 \le i \le j \le |C_l|} dist(x_i, x_j)$$

  - The distance between nearest samples of clusters C_i and C_j

$$d_{min}(C_i, C_j) = min_{x_i \in C_i, x_j \in C_j} dist(x_i, x_j)$$

  - Distance between centers of clusters C_i and C_j

$$d_{cen}(C_i, C_j) = dist(\mu_i, \mu_j)$$

- Davies-Bouldin Index (DBI)

$$DBI = \frac{1}{k} \sum_{i=1}^{k} \max_{j \neq i} \left(\frac{avg(C_i) + avg(C_j)}{d_{cen}(\mu_i, \mu_j)} \right)$$

- Dunn Index (DI)

$$DI = \min_{1 \le i \le k} \left\{ \min_{j \neq i} \left(\frac{d_{min}(C_i, C_j)}{\max_{1 < l < k} diam(C_l)} \right) \right\}$$

距离的计算方式需要满足：

- Nonnegativity: $dist(x_i, x_j) \geq 0$
- Identity: $dist(x_i, x_j) = 0$, if and only if $x_i = x_j$
- Symmetry: $dist(x_i, x_j) = dist(x_j, x_i)$
- Directness: $dist(x_i, x_j) \leq dist(x_i, x_k) + dist(x_k, x_j)$

通常使用范数或者余弦相似度。

14.2 层次聚类算法HAC

聚类从N个组开始，每个组最初包含一个对象，然后每一步合并两个最相似的组，直到有一个包含所有数据的组出现。

Algorithm 11: Agglomerative clustering

Initialize clusters as singletons: **for** $i \leftarrow 1$ **to** n **do** $C_i \leftarrow \{i\}$;
;
Initialize set of clusters available for merging: $S \leftarrow \{1, \ldots, n\}$; **repeat**
    Pick 2 most similar clusters to merge: $(j, k) \leftarrow \arg\min_{j,k \in S} d_{j,k}$;
    Create new cluster $C_\ell \leftarrow C_j \cup C_k$;
    Mark j and k as unavailable: $S \leftarrow S \setminus \{j, k\}$;
    if $C_\ell \neq \{1, \ldots, n\}$ **then**
        Mark ℓ as available, $S \leftarrow S \cup \{\ell\}$;
    foreach $i \in S$ **do**
        Update dissimilarity matrix $d(i, \ell)$;
until *no more clusters are available for merging*;

三种HAC，取决于如何定义簇之间的差异：

单链接：两个簇之间的最近距离。

完整链接：两个簇之间的最远距离

平均链接：两个簇内所有对象组的距离的平均值。

14.3 k-means

一个簇的中心定义为簇内所有点到该中心平均距离最近的中心点。

对于欧氏距离，中心是簇内样本的均值。

k-means的目标是最小化平方误差（所有样本到该簇中心点距离平方的和）。

k-means的算法

随机选择k个样本作为初始均值向量。

计算每个样本与均值向量的距离，选择最近的作为自己的所属簇。

更新均值向量。

循环直到当前均值向量均为更新。

初始均值向量选取

先均匀随机选取初始点，再从剩余的点中选择后续点。其概率与它到最近的初始点的平方距离成正比。

$$p(\boldsymbol{\mu}_t = \mathbf{x}_n) = \frac{D_{t-1}(\mathbf{x}_n)^2}{\sum_{n'=1}^{N} D_{t-1}(\mathbf{x}_{n'})^2}$$

$$D_t(\mathbf{x}) = \min_{k=1}^{t-1} ||\mathbf{x} - \boldsymbol{\mu}_k||_2^2$$

如何选择k？采用**轮廓系数**。

$$sc(i) = (b_i - a_i)/\max(a_i, b_i)$$

a是平均类内距离，b是到最近簇的平均距离。

把所有样本的轮廓系数均值作为一个k类聚类的轮廓系数。

15 聚类2

15.1 谱聚类

将数据点作为图的顶点，每对顶点都有一条边连接。边的权重大表示相邻顶点相似程度大。

目标是最小化损失函数：

$$Ncut(A, B) = \frac{cut(A, B)}{\sum_{i\in A,j} W_{ij}} + \frac{cut(B, A)}{\sum_{i\in B,j} W_{ij}}$$

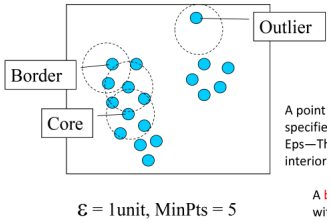
$$cut(A, V - A) = \sum_{i\in A, j\in V-A} W_{ij}.$$

15.2 基于密度的分类DBSCAN

核心点：在ϵ邻域内有多于MinPts个点。

边界点：在ϵ邻域内有少于MinPts个点，但是在核心点的邻域内。

噪声点：不算核心点或边界点。



密度直达：q是从p密度直达的，如果p是核心点，q是p的邻居。

密度可达：q是从p密度直达的，如果p到p1密度直达、p1到q密度直达。

```
for each o ∈ D do
    if o is not yet classified then
        if o is a core-object then
            collect all objects density-reachable from o and assign
            them to a new cluster.
        else
            assign o to NOISE
```

优点

抗噪音。

可以处理不同形状和大小的聚类。

缺点

无法处理不同的密度。

对参数敏感，难以确定正确参数集。

15.3 高斯混合聚类

假设每个簇都由各自的概率分布生成，则点的生成是，随机选择一个簇，再根据概率分布生成一个点给这个簇。

则高斯混合模型的参数有：簇的选中概率，每个簇对应概率分布的均值和方差。

$$\begin{aligned} p(x,z) &= p(z)p(x \mid z) \\ &= \pi_z \mathcal{N}(x \mid \mu_z, \Sigma_z) \end{aligned}$$

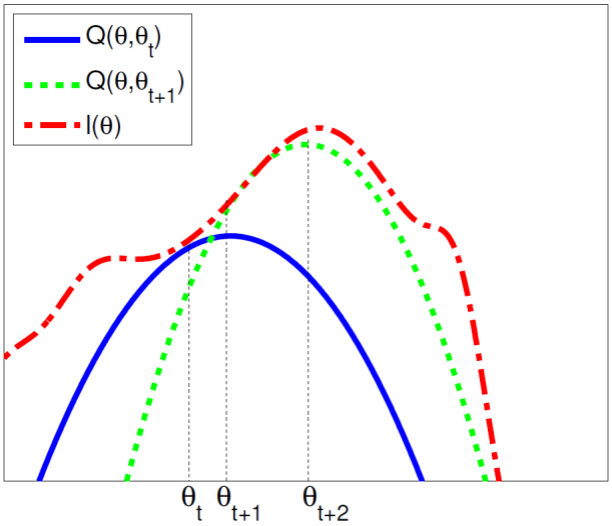
z是隐变量(生成样本x_i的高斯混合成分，未知的)，我们通过观察x，通过后验概率用最大似然估计θ的值。

$$\begin{aligned} L(\pi,\mu,\Sigma) &= \prod_{i=1}^{n} p(x_i) \\ &= \prod_{i=1}^{n} \sum_{z=1}^{k} \pi_z \mathcal{N}(x_i \mid \mu_z, \Sigma_z). \end{aligned}$$

$$J(\pi,\mu,\Sigma) = \sum_{i=1}^{n} \log \left\{ \sum_{z=1}^{k} \pi_z \mathcal{N}(x_i \mid \mu_z, \Sigma_z) \right\}$$

16 EM算法

16 .1 边界优化



目标是找到$L(\theta)$的最大值。

找到一个便于计算的代理函数$Q(\theta)$逼近$L(\theta)$的下界，并且在θ^t处值相等。

找到$Q(\theta)$的最大值，过最大值找一个新的代理函数，再找它的最大值，以此类推。

16.2 EM算法

是边界优化的特殊情况。

X：已观测变量。Z：隐变量。Θ：模型参数。

目标是最大化似然$P(X,Z|\Theta)$

E：根据当前的参数Θ^t推断隐变量分布$P(Z|X,\Theta^t)$，并计算对数似然$LL(\Theta|X,Z)$关于Z的期望$Q(\Theta|\Theta^t)$。【利用当前估计的参数值，计算对数似然的期望值】

M：寻找参数最大化期望似然：$\Theta^{t+1} = \arg\max_{\Theta} Q(\Theta|\Theta^t)$。【寻找能使E步产生的似然期望最大化的参数值，将新的参数值重新用于E步】

【重复至收敛到局部最优解】

16.3 课堂测试

什么时候使用EM算法？有隐变量的情况。

EM算法是聚类算法吗？不是，是可以用于解决聚类问题的一种优化算法。

EM算法是监督还是无监督方法？无监督。

EM能确保在迭代中不让目标函数的值降低吗？可以，因为每次都取当前条件下最优的Θ。

K-means和EM算法的关系？K-means用到了EM的思想，两个步骤分别对应。

17 半监督学习1

同时使用有标签和无标签的数据来进行学习。

17.1 自训练

假设高置信度的预测是对的，则有自训练算法：

先通过有标签的数据集训练，再对无标签的样本进行测试，把得到标签的新样本也加入训练集，再训练，重复。

可以选择加置信度最高的少部分数据，加全部，或者按照置信度加权的加全部。

优点

简单。

可以用于不同的模型。

缺点

早期的错误会强化它们自己。

不一定收敛。

17.2 生成模型

GMM：最大化$p(X_l, Y_l, X_u|\theta) = \sum_{Y_u} p(X_l, Y_l, X_u, Y_u|\theta)$

优点

清晰的概率框架，当模型接近正确解时很有效。

缺点

局部最优。

如果生成模型出现错误，未标记的数据可能会造成伤害。

17.3 转导SVM

Objective function of transductive support vector machine (TSVM):

$$
\begin{aligned}
\min_{\boldsymbol{w},b,\hat{\boldsymbol{y}},\boldsymbol{\xi}} \quad & \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C_l\sum_{i=1}^{l}\xi_i + C_u\sum_{i=l+1}^{m}\xi_i \\
\text{s.t.} \quad & y_i(\boldsymbol{w}^\top\boldsymbol{x}_i + b) \geq 1-\xi_i, \quad i = 1,\ldots,l, \\
& \hat{y}_i(\boldsymbol{w}^\top\boldsymbol{x}_i + b) \geq 1-\xi_i, \quad i = l+1,\ldots,m, \\
& \xi_i \geq 0, \quad i = 1,\ldots,m,
\end{aligned}
\tag{1}
$$

先用有标签的样本学习一个SVM，再用这个SVM对无标签数据进行标记指派，也就是把预测结果作为伪标签。把这些数据也加入训练集，训练得到新的SVM。找出两个标记指派为异类而且可能发生错误的无标签样本，交换标签。再重新训练SVM，重复。

输入：有标记样本集 $D_l = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_l, y_l)\}$;
未标记样本集 $D_u = \{\boldsymbol{x}_{l+1}, \boldsymbol{x}_{l+2}, \ldots, \boldsymbol{x}_{l+u}\}$;
折中参数 C_l, C_u.

过程：

用 D_l 训练一个 SVM_l;
用 SVM_l 对 D_u 中样本进行预测, 得到 $\hat{\boldsymbol{y}} = (\hat{y}_{l+1}, \hat{y}_{l+2}, \ldots, \hat{y}_{l+u})$;
初始化 $C_u \ll C_l$;
while $C_u < C_l$ **do**
  基于 $D_l, D_u, \hat{\boldsymbol{y}}, C_l, C_u$ 求解式(13.9), 得到 $(\boldsymbol{w}, b)$, $\boldsymbol{\xi}$;
  while $\exists\{i, j \mid (\hat{y}_i\hat{y}_j < 0) \wedge (\xi_i > 0) \wedge (\xi_j > 0) \wedge (\xi_i + \xi_j > 2)\}$ **do**
    $\hat{y}_i = -\hat{y}_i$;
    $\hat{y}_j = -\hat{y}_j$;
    基于 $D_l, D_u, \hat{\boldsymbol{y}}, C_l, C_u$ 重新求解式(13.9), 得到 $(\boldsymbol{w}, b)$, $\boldsymbol{\xi}$
  end while
  $C_u = \min\{2C_u, C_l\}$
end while

输出：未标记样本的预测结果: $\hat{\boldsymbol{y}} = (\hat{y}_{l+1}, \hat{y}_{l+2}, \ldots, \hat{y}_{l+u})$

---

18 半监督学习2

18.1 基于图的算法

数据为顶点，边的权重表示连接的两端为同标签的可能性（即相似度）。通过图把有表情数据的标签传递给无标签数据。

在每次迭代中，给定数据点的标签分布被计算为其所有连接数据点的标签分布的加权平均值，其中权重对应于T中的边缘权值。

18.2 Multiview：协同训练

把有标签数据集分成两份，分别训练两个分类器。

分别拿两个分类器预测无标签数据，把最高置信度的部分数据作为有标签数据给另一个数据集，再进行训练。

优点

适用于不同的分类器。

对错误的敏感性比自训练底。

18.3 课堂测试

什么是自训练？它的优缺点？

什么是半监督的生成模型，为什么有用？

TSVM如何工作？

为什么基于图的半监督有用？它的关键是什么？

什么是半监督的multiview算法？

半监督中自训练和协同训练的区别？

19 相似搜索

19.1 Locality-sensitive Hashing (LSH)

- Goal: finding similar documents
- Document representation

Documents

Words

1	1	1	0
1	1	0	1
0	1	0	1
0	0	0	1
1	0	0	1
1	1	1	0
1	0	1	0

- Each column represents a document
- '1' represents the existence of corresponding word

- Measuring the similarity between documents A and B
  - $A \cap B$ denotes the common words between documents A and B
  - $A \cup B$ denotes all of the words in documents A and B
  - Similarity between A and B can be measured as

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

where $|\cdot|$ is size of set; $J(\cdot,\cdot)$ is also called the Jaccard index（雅卡尔指数）

【感觉不会考随便吧】

复习课

Q1 深度学习和机器学习的区别?

机器学习学习结构化数据，并以此来进行决策。深度学习则可以自己处理数据，并且像人脑一样识别，分析，再做决策。

主要区别：

- 机器学习需要结构化数据、深度学习依赖于人工神经网络层。

Q2 监督和无监督学习的区别?

前者使用有标签数据，后者使用无标签数据。

Q3 如何再一个数据集中选择重要的变量

- 去掉一些依赖于其他变量的变量。
- Lasso回归。
- 随机森林。
- 根据信息增益。

Q4 给定一个数据集，如何选择算法

如果数据是线性的，则用线性回归，如果不是线性的，则用bagging/kernel算法。

如果需要用于商业分析之类的，用决策树，SVM。

如果输入是图片，视频，音频，则适合神经网络。

Q5 正则化在机器学习中的作用

避免过拟合，降低模型复杂度。

Q6 数据高方差是好的吗

高方差意味着数据展开范围较大，数据具有多样性。通常不是好事。

Q7 如果数据集高方差，应该怎么处理

使用bagging算法。

Q8 梯度下降和随机梯度下降的区别

梯度下降使用所有数据更新参数、随机梯度下降则随机选择一个数据更新参数。

Q9 决策树的优缺点

优点：易解释性，对异常点比较鲁棒，需要调的参数较少。

缺点：容易过拟合。

Q10 混淆矩阵是什么，有什么用

混淆矩阵是一个用于衡量分类模型性能的表格。

Q11 解释维度诅咒

当数据集有过多属性时，容易过拟合，并且对于一些不重要的属性会同样关注，以影响结果。还会导致计算时间上升。

Q12 正则化和标准化的区别

标准化调整数据，当数据的范围很大时进行标准化，以减少精度损失。

正则化调整预测函数，以避免过拟合和噪音影响。

Q13 Normalization和Standardization的区别

前者是把数据放大缩小规模到0到1之间。

后者是把数据放大缩小至均值为0，标准差为1。

Q14 回归和分类的区别

回归和分类都是监督学习。

回归的输出是数字（连续），分类的输出是类别（离散）。

Q5 什么算法被称为lazy learner

KNN。因为他不学习任何数据集信息，只是单纯进行计数。

Q16 KNN， K-means区别

前者分类，后者聚类。

Q17 SVM的核函数作用

把线性不可分的数据映射到高维空间使之线性可分。

Q18 什么是集成模型？解释集成技术如何比传统机器学习方法更优

集成是一组模型一起用于预测。因为混合了多个模型，因此比起单个模型有更好的预测性能，比如减少方差、减少偏差、减少过拟合。

Q19 什么是过拟合，欠拟合，决策树为什么会过拟合

过拟合是一个模型容易被数据的噪音影响，欠拟合是不足以拟合数据，表现出低方差高偏差。

决策树中，树完美拟合训练集中的所有数据，分支对数据的判断规则非常严格，因此对于非训练集的数据，准确率受到影响。

Q20 解释Lasso和岭回归区别

前者用L1、后者用L2。弹性网络结合二者。

Q21 概率和似然的区别

概率衡量一个事件发生的可能性。

似然描述对于特定参数空间，根据观察数据，参数取某个值的可能性。

Q22 如何剪枝

预剪枝，后剪枝。

Q23 模型准确性和性能的区别

性能包括了准确性等等。

Q24 如何处理不平衡的数据集

过采样，欠采样。

Q25 特征工程的重要性

提取特征，降低模型复杂度，提升模型准确性。

Q26 Boosting Bagging区别

前者串行，后者并行。

Q27 生成模型和判别模型的区别

前者基于联合概率找分布，后者基于条件概率找判断边界。

Q28 超参数和参数的区别

超参数不可训练，参数可训练。

Q**29 如何决定聚类算法中簇的个数**

用轮廓系数。

Q30 交叉验证

交叉验证用于提升机器学习算法性能，把数据集划分成等大的多个组，其中随机的部分作为测试集，剩余部分作为训练集。

Q31 随机森林用了什么集成学习技术

Bagging。

Q32 熵和信息增益的区别

信息增益基于数据集基于一个属性分解时熵减的程度。

Q33 使用SVM而不用随机森林的例子

线性可分，维度较高时用SVM。

Q34 SVM受到异常点影响?

C很大的话会，反之不会。

Q35 可以在逻辑回归中用核函数吗

逻辑回归的计算代价远比SVM高，因为需要考虑所有点。

Q36 SVM能给出概率输出吗

不能直接给出，但是可以通过五折交叉验证给出。

Q37 贝叶斯风险和经验风险区别

前者是理想情况的风险，后者是通过数据集得出的风险。

Q38 什么是拉格朗日乘子

在最优化中，把原优化问题转换为对偶问题时，引入拉格朗日乘子作为乘以限制项的参数。

Q39 随机森林的随机性

数据随机，属性随机。

Q40 为什么随机森林不容易过拟合

因为使用bagging，对于数据噪音不敏感。

Q41 深度学习，集成学习区别

前者类似CNN，Transformer之类的，用到了集成的思想。

Q42 集成的分类器是否一定比单个基分类器更准确?

不一定。