CPU EDUCATION

COMP9417 公开课

Week1 公开课讲解

TUTOR：Ayr 学姐　　时长：2 小时

本周内容预览：

1. 个人介绍
2. 课程 Overview (包括考核介绍)
3. CPU 课程福利
4. 线性回归
5. 梯度下降
6. 方差-偏差权衡（是重点，但不是本周的重点）
7. 历年相关考题讲解

CPU EDUCATION

关于CPU EDUCATION

CPU Education成立于2018年，总部在澳大利亚墨尔本，目前国内办公室在广州。经过近多年的积累与发展，公司凭借优质的教学质量和专业的研发团队力量不断壮大，成为全球最受学生欢迎的教育机构之一，在众多学生及老师中享有良好的口碑。CPU Education争做行业标杆，为不同阶段的学生提供专业，专属，专注的教学服务。

致力于为学生实现人人HD的目标，以“为学生提供极具价值的教学服务，培训各行业精英”为宗旨。秉承“教育高于一切”的经营理念，本着服务定制化、师资专业化、教学规范化的要求，打造墨尔本教育行业的“黄埔军校”。

课后，如果您有任何建议和意见，我们都非常欢迎您联系小助手分享您的想法，给予我们改进和提高的机会!
感谢您参与CPU Education的辅导课程!

课程体验反馈

PART 1 个人介绍

- Name：Ayr
- Bachelor of Mathematics (NEU)
- Master of Data Science (USYD)
- 全部课程分数均 84，机器学习相关的课全部 85+
- 本科数学，算法和统计强
- 山东考生，应试技能 MAX

PART 2 课程介绍

➢ 机器学习中的核心算法和模型类型 -- 如决策树、支持向量机、KNN 等

➢ 关于从数据中学习的基础概念 -- 训练、测试、验证的过程，以及数据预处理等

➢ 相关理论，用于深化和概括理解 -- 如泛化误差、偏差-方差权衡

➢ 实际应用 -- 代码

X 概率和统计（需要具备基础知识）--不会考很难的数学知识

X 大量的神经网络和深度学习、强化学习、大数据

- Week 1: Regression
- Week 2-3: Classification、
- Week 4-5: Tree-based method
- Week 6: 核方法、SVM
- Week 7: 集成
- Week 8: 神经网络
- Week 9: 无监督学习 - 聚类、PCA
- Week 10: 学习理论


ELLE学姐


ECHO学姐

PART 3 考试介绍

- 考试时间为 2 小时，in-person exam，占总分的 50%
- 两个 homework，分别在第四周和第七周，各占 15%
- 小组项目，第十周，占 20%
- 这门课的 Final 通常不会考代码，主要考察算法和理论
- 题型涵盖选择题，简答题和计算题
- 同学普遍反应期末难度较高
- 可能会考伪代码

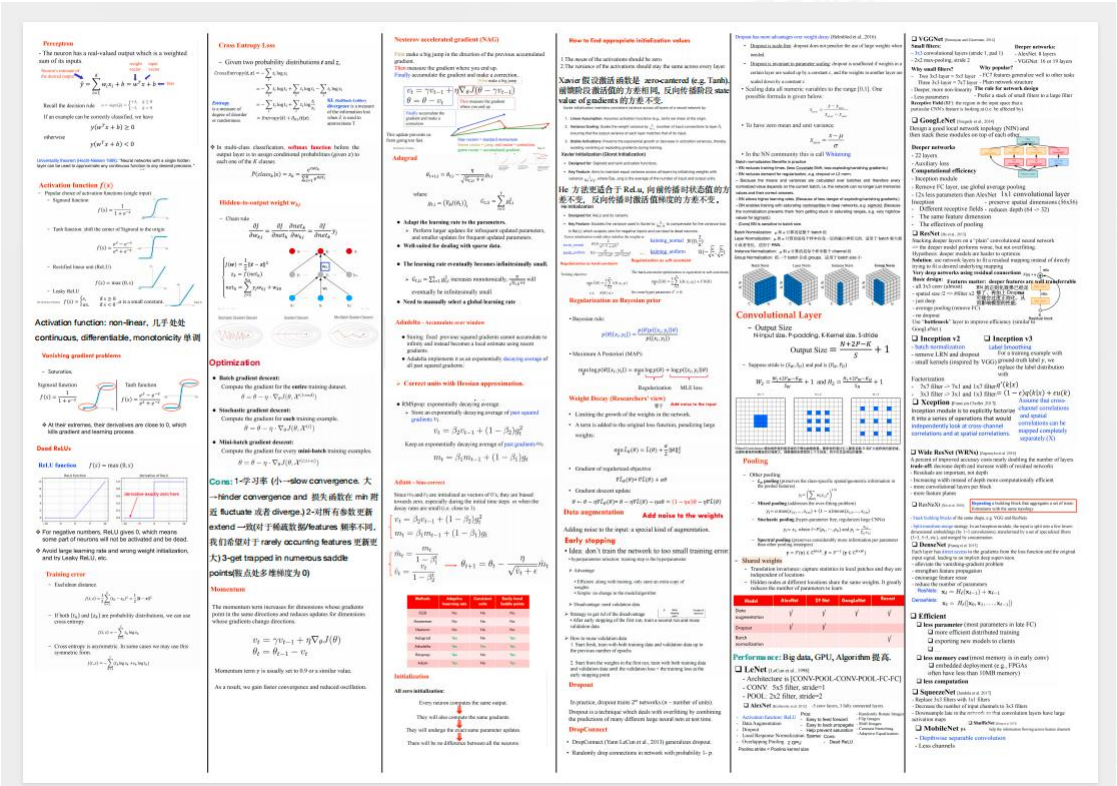
PART 4 作业介绍

- 两个 homework - 各占 15%，都是单人完成 - due week 4, week 7
- 一个 project - 占 20%，4-5 人,不强制一个 tut 组队, week9, 10
- 难度比较高
- 去年的 homework 1 是梯度下降的推导、分析和计算，并且需要写代码
- homework 2 是 Newton 方法和逻辑回归问题的梯度下降与对偶形式优化，要求从零推导更新公式并实现代码。
- project 是实现一个完整的图像分类的任务
- 两个 homework 对于数学统计知识的要求比较高

PART 5 CPU 平时班课堂福利

- 精讲：双语细致的讲解 slides 中的内容，并且去掉不考的内容，对于英语或者是数学背景不够的同学很友好

- 刷题：每周的知识点都会配合往年的 quiz 和考题去讲解，让大家理解知识点、并且会做题
- 押题：将会结合过往考题进行本学期的押题，预测题会在期末班中讲解，往年押题命中率高达 80%
- 答疑：报名平时班的同学会有一个答疑群，有不懂的问题在群里问，我看到了就会回答
- 笔记：课后会把上课的讲义上传，供大家学习参考
- 考前 cheat sheet：考前报名期末班的同学会有专属的 cheat sheet 的参考



Any Question?


ELLE学姐


ECHO学姐

一、课程介绍

1. What is Machine Learning?

机器学习是人工智能的一个分支，其主要目标是通过算法和统计模型，使计算机系统能够从数据中学习，而无需明确地为每一项任务编程。机器学习主要分为三种类型：**监督学习(supervised learning)、无监督学习(unsupervised learning)和强化学习(reinforcement learning)。**

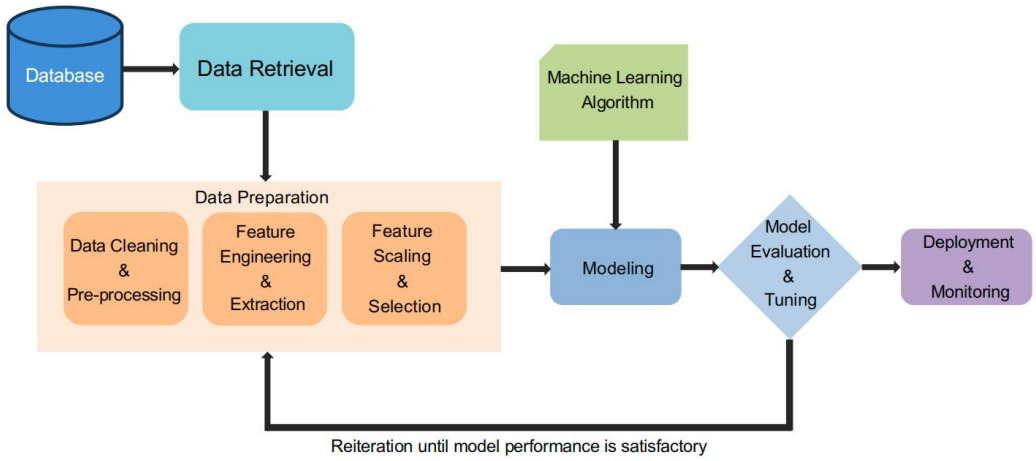
机器学习就像教一个学生解决问题，但不是直接告诉他答案，而是给他很多例子，通过这些例子他学会自己去总结规律。比如：教计算机识别猫的照片，不是手写“这是猫”的规则，而是给它大量猫和非猫的图片，让它自己总结“猫”的特征。

2. What is Data Mining?

数据挖掘是从大量数据中提取有用模式和知识的过程，涉及统计学、数据库技术、机器学习等领域。其目的是从数据集中提取隐含的、之前未知的信息

数据挖掘就像在一座金矿里淘金，面对海量的数据，找到隐藏的、有价值的信息。
比如：通过分析超市的销售数据，发现“买啤酒的人经常会买尿布”，然后调整货架摆放，增加销售额。

3. 机器学习的 pipeline (Assignment 会用到)



a) Database 数据库 - 想象我们有一个巨大的数据库，存储了用户的观影记录，比如谁看了什么电影、评分多少。

b) Data retrieval 数据检索 - 从数据库中提取所需数据。这一步通常涉及连接数据源、查询数据、获取目标数据集 --非重点 - 提取用户 A 的电影评分数据，以及他没有看过的电影列表。

c) 数据准备（Data Preparation）

d) 数据清理与预处理（Data Cleaning & Pre-processing）：处理缺失值、异常值、数据重复等。如果有用户给电影评分为"-100"，那就清理掉这种异常数据。

e) 特征工程与提取（Feature Engineering & Extraction）：设计和生成有助于模型训练的特征，例如 feature transformation, feature interaction, Encoding 等。 例子：从数据中提取有用的特征，比如用户的观影习惯（喜欢科幻片、动画片等）

f) 特征缩放与选择（Feature Scaling & Selection）：数据标准化、归一化、删除冗余特征。例如：统一数据格式，比如把评分范围调整为 0 到 1 之间

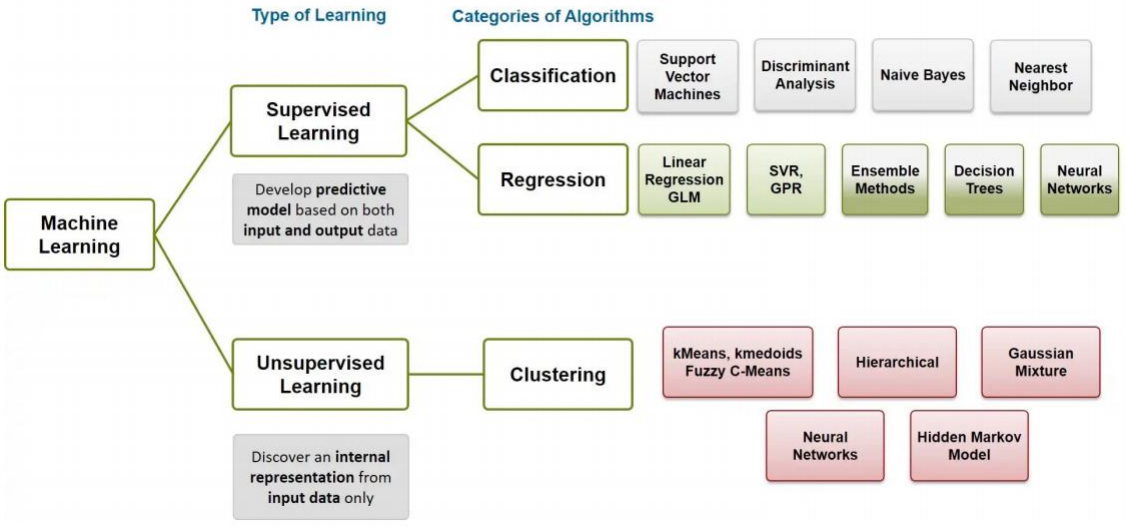
g) Model 模型和算法 (重点) - 使用机器学习算法训练模型，让系统学会根据用户的喜好推荐电影。

h) 模型评估与调优（Model Evaluation & Tuning）- 检查模型的准确性，如果推荐的电影不符合用户兴趣，就调整参数或选择更好的算法。

i) 部署与监控（Deployment & Monitoring: 应用到实际场景中 -- 非重点 模型上线，开始实时推荐，并监控效果。

j) 迭代优化：如果表现不够好？ 不断更新数据、优化模型，直到满意为止。比如随着用户 A 打分更多的电影，系统逐渐学会推荐更精准的内容。

二、监督学习 VS 无监督学习:

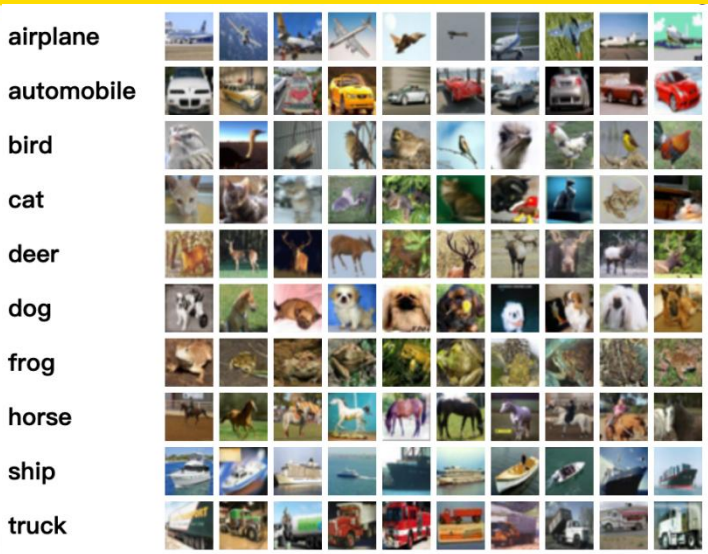


1. 监督学习：有 label 无监督学习：没有 label

2. 什么是 label?

在机器学习中，label（标签）是指数据样本的真实分类或目标值，模型通过学习标签与特征之间的关系来进行预测。直观的理解，label 就是你希望模型预测的答案，比如图像识别中的“猫”或“狗”、房价预测中的具体价格。

e.g. The CIFAR-10 dataset :



	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33	36.2

监督学习的两种类型：

分类 -- label 是 categorical

回归 -- label 是 numerical

无监督学习：聚类

三、线性回归

1. 一元线性回归

二元一次方程，我们将 y 作为因变量，x 作为自变量，得到方程：

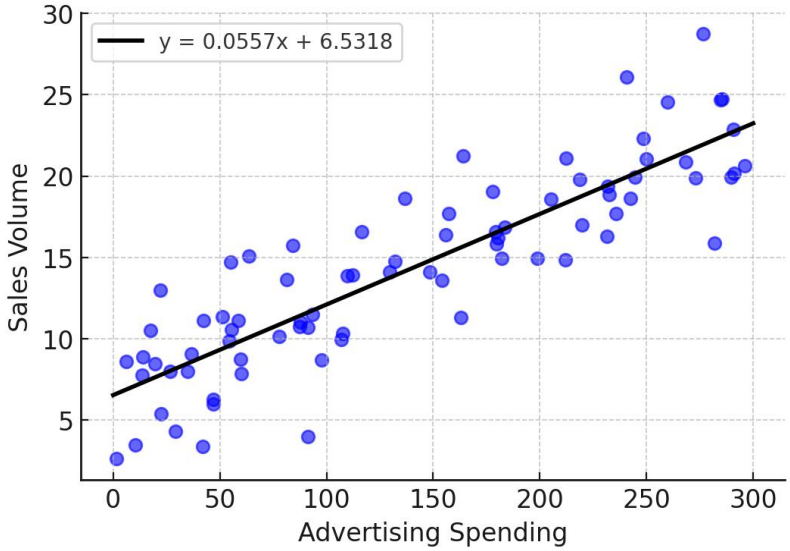
$$y = \beta_0 + \beta_1 x$$

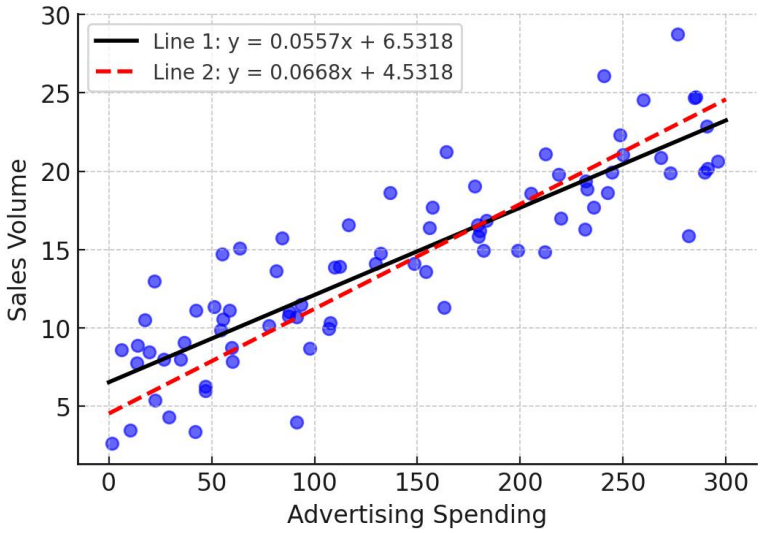
当给定参数β_0和β_1的时候，画在坐标图内是一条直线（“线性”）

当我们只用一个 x 来预测 y，就是一元线性回归，也就是在找一个直线来拟合数据。

例如，图中横坐标代表广告投入金额，纵坐标代表销售量，**线性回归就是要找一条直线，并且让这条直线尽可能地拟合图中的数据点。**

那既然是用直线拟合散点，为什么最终得到的直线是 y = 0.0557x + 6.5318，而不是下图中的 y = 0.0668x + 4.5318 呢？这两条线看起来都可以拟合这些数据啊？毕竟数据不是真的落在一条直线上，而是分布在直线周围，所以我们要找到一个评**判标准，用于评价哪条直线才是最“合适”的。**



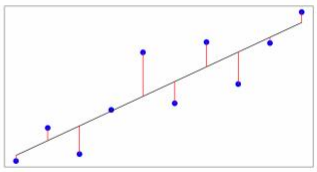


2. 什么是误差?

我们得到的拟合方程是 y = 0.0517x + 6.5318，此时当我们获得一个新的广告投入金额后，我们就可以用这个方程预测出大概的销售量。

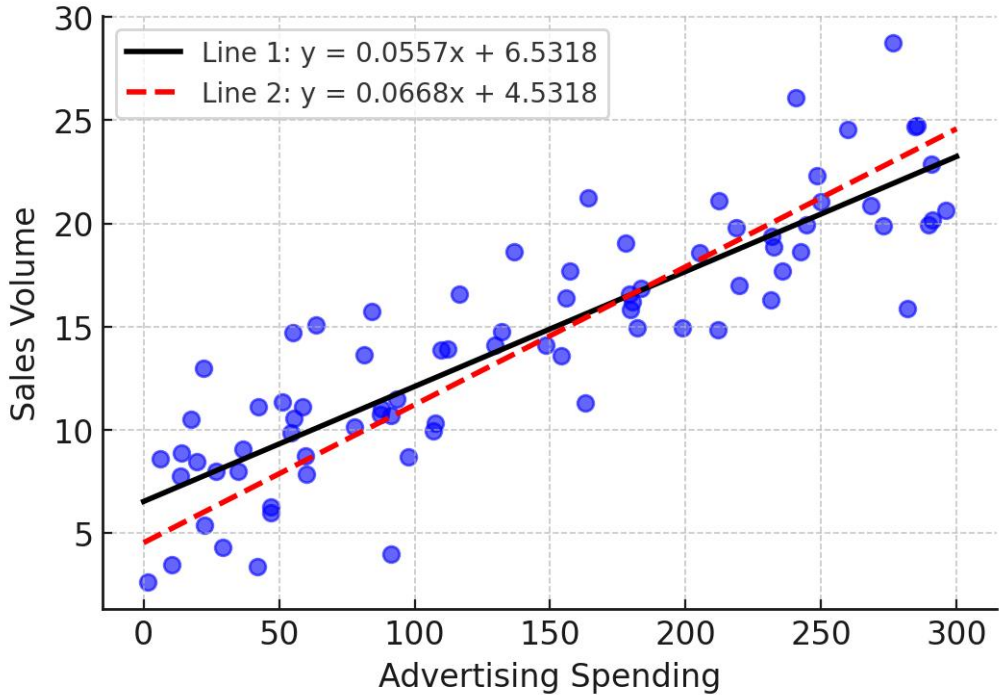
代入 x=0，我们可以得到一个唯一的$\hat{y} = 6.5318$，这个$\hat{y}$不是我们真实观测到的，而是估计值

Error = Difference between the predicted value and the actual value



We want to minimize the error over all samples!

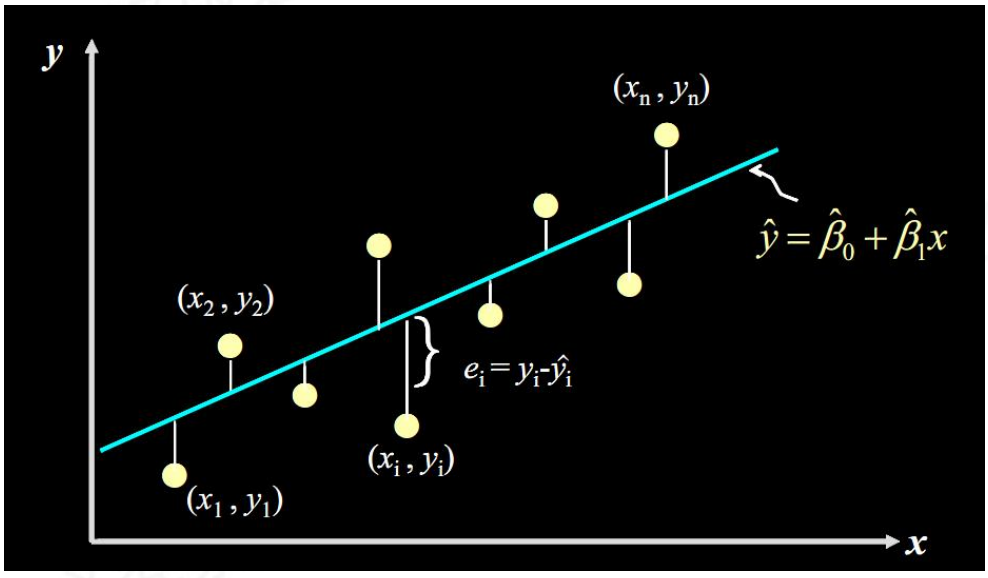
3. 损失函数



哪条直线才是最"合适"的?

Residual: 真实值和预测值的差距 -- 观测误差，基于样本数据计算

$$e = y - \hat{y}$$



RSS -- Residual Sum of Squares/Sum of Squared Errors 残差平方和

$$J(\theta) = \sum_{j=1}^{m}(y_j - \sum_{i=0}^{n}\theta_i x_{ji})^2 = \sum_{j=1}^{m}(y_j - x_j^T\theta)^2 = (\mathbb{y} - X\theta)^T(\mathbb{y} - X\theta)$$

$$RSS = \sum_{i=1}^{m}(y_i - \hat{y}_i)^2$$

损失函数是衡量回归模型误差的函数，也就是我们要的“直线”的评价标准。这个函数的值越小，说明直线越能拟合我们的数据。

还有一种常用的损失函数是均方误差 **MSE(Mean squared error)**：

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2$$

那β0和β1的具体值究竟是怎么算出来的呢?

四、最小二乘法（Ordinary Least Squares，部分掌握）

单变量回归 vs 多变量回归

- **Univariate regression**: one input variable/feature/attribute is used to predict one output variable
- **Multiple regression / multivariable regression**: more than one variables/features/attributes are used to predict one output variable

扩展到多元情况，多元线性回归方程的一般形式为：

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$$

可以简写为矩阵形式（一般加粗表示矩阵或向量）：

$$\boldsymbol{Y} = \boldsymbol{X\beta}$$

其中，

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & \dots & x_{1n} \\ 1 & x_{21} & x_{22} & \dots & x_{2n} \\ & & \vdots & & \\ 1 & x_{m1} & x_{m2} & \dots & x_{mn} \end{bmatrix} \qquad \mathbb{y} = \begin{bmatrix} y_1 \\ y_2 \\ \\ y_m \end{bmatrix}$$

其中，X 是特征矩阵，其中每一行对应一个样本，每一列对应一个特征。
矩阵 X 第一列的全 1 是为了表示截距项（bias）。

y 是目标变量的列向量。
注意事项： 必须在 X 矩阵中添加一列全为 1 的列，用于计算截距参数。

闭式解法：通过显式求导并令导数为零，找到 J(θ) 的最小值

$$J(\theta)=\sum_{j=1}^{m}(y_j-\sum_{i=0}^{n}\theta_i x_{ji})^2=\sum_{j=1}^{m}(y_j-x_j^T\theta)^2=(\mathbb{y}-X\theta)^T(\mathbb{y}-X\theta)$$

$$\frac{\partial}{\partial\theta}J(\theta)=0$$

$$J(\theta)=(\mathbb{y}-X\theta)^T(\mathbb{y}-X\theta)$$

$$\frac{\partial}{\partial\theta}J(\theta)=-2X^T(\mathbb{y}-X\theta)=0$$

$$X^T(\mathbb{y}-X\theta)=0$$

$$\boxed{\theta=(X^TX)^{-1}X^T\mathbb{y}}$$

这个地方记住最后的公式就可以，考试不会考推导的，仅了解即可
但只要满足没有完美多重共线性，线性回归的最小二乘法就有解

完美共线性：特征矩阵 X 的列之间存在严格的线性相关关系，即某一列可以由其他列的线性组合精确表示

$$x_i=\sum_{j\neq i}\alpha_j x_j$$

当存在完美多重共线性时，线性回归模型的解

$$\hat{\beta}=(X^TX)^{-1}X^Ty$$

中的 $(X^TX)^{-1}$ 不存在，因为 X^TX 是不可逆的，也就是说 X^TX 是满秩矩阵

Question: 线性回归会受高维数据的影响吗？(p>>n)

怎么去拟合线性回归?
如果我矩阵我算不出来呢?
-- 数值解 梯度下降 Gradient Descent

五、梯度下降 Gradient Descent

梯度 -- 偏导

为什么用梯度下降?
-- 直接求导计算复杂度大 $O(mn^2+n^3)$，梯度下降单次迭代复杂度 $O(mn)$
-- 即使线性回归没有解析解，仍然可以使用梯度下降来求解参数

梯度下降，就是通过一步步迭代，让所有偏导函数都下降到最低。


ELLE学姐


ECHO学姐

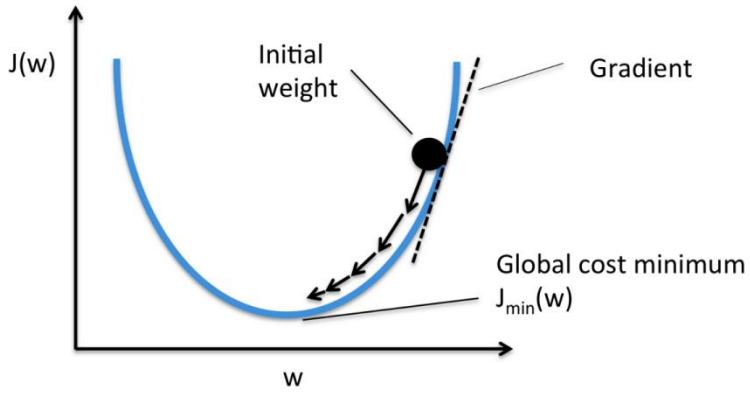
先初始化一个点，也就是随便选择一个位置，计算它的梯度，然后往梯度相反的方向，每次移动一点点，直到达到停止条件。

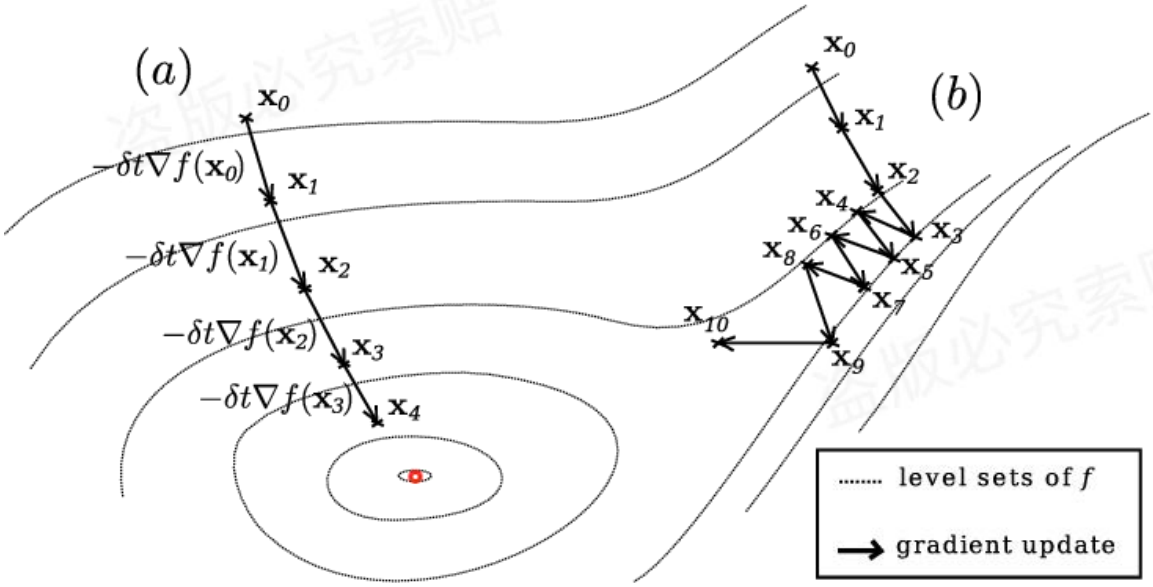
这个停止条件，可以是**足够大的迭代步数**，也可以是一个比较**小的阈值**，当两次迭代之间的差值小于该阈值时，认为梯度已经下降到最低点附近了。

梯度下降的参数更新公式：

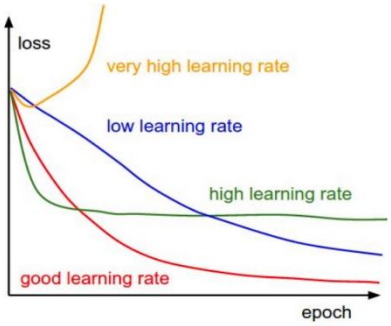
$$\theta_i^{(t+1)} := \theta_i^{(t)} - \alpha \frac{\partial}{\partial \theta_i} J(\theta_i^{(t)})$$

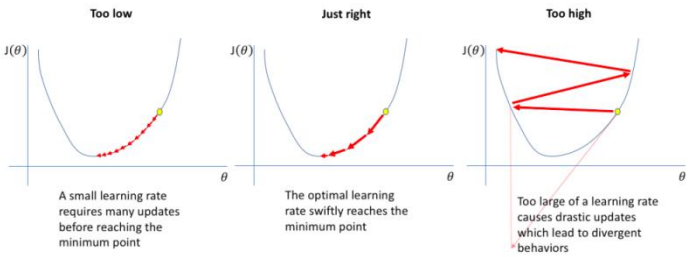
α 是学习率（Learning Rate），控制每次参数更新的步长。
梯度方向是使损失函数 J(θ) 减少最快的方向。





学习率





如果 α 太小，算法收敛速度慢。
如果 α 太大，可能导致算法无法收敛，甚至发散。

以线性回归 $h_\theta(X) = X\theta$ 为例，线性回归的损失函数为

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x_i) - y_i)^2$$

矩阵表示为

$$J(\theta) = \frac{1}{2m}(X\theta - Y)^T(X\theta - Y)$$

算法过程：

1. 初始化θ_0 向量的值
2. 计算损失函数的梯度

$$\nabla_\theta J(\theta) = \frac{1}{2m}\cdot 2X^T(X\theta - Y)$$

化简得到，

$$\nabla_\theta J(\theta) = \frac{1}{m}\cdot X^T(X\theta - Y)$$

3. 将其代入 $\nabla_\theta J(\theta)$ 得到当前位置的梯度$\nabla_\theta J(\theta_0)$;

4. 更新参数$\theta_1 = \theta_0 - \alpha\nabla_\theta J(\theta_0)$

5. 重复以上步骤，直到更新到某个 θ_k，达到停止条件，这个 θ_k就是我们求解的参数向量。

梯度下降分类

上式里面，我们使用的是**批量梯度下降（Batch Gradient Descent, BGD）**，每次使用整个训练数据集 X 来计算梯度，然后一次性更新参数

1. **Batch Gradient Descent:**

$$\theta_i^{(t+1)} = \theta_i^{(t)} + \alpha \frac{2}{m} \sum_{j=1}^{m} (y_j - h_{\theta^{(t)}}(x_j)) x_{ji} \quad \text{(for every } i\text{)}$$

Replace the gradient with the sum of gradient for all samples and continue until convergence.

Convergence means that, the estimated θ will be stabilized.

计算量较大：每次迭代都要计算整个数据集的梯度。
稳定收敛：梯度方向准确，下降过程稳定。
适用于小型数据集：如果数据量很大（如上亿条数据），批量计算梯度的代价会很高。

2. **Stochastic Gradient Descent:**

$$for\ j = 1\ \ to\ \ m\{$$
$$\theta_i = \theta_i + 2\alpha(y_j - h_\theta(x_j))x_{ji} \quad \text{(for every } i\text{)}$$
$$\}$$

Repeat this algorithm until convergence.

随机梯度下降（SGD, Stochastic Gradient Descent） 则是在每次迭代时随机选取一个样本来计算梯度并更新参数，而不是像批量梯度下降（BGD）那样使用整个数据集。

计算更快：每次迭代只使用**一个样本**来计算梯度，适用于大规模数据集。
更快收敛：更新频率更高，可能在更少的迭代次数内找到较优解。

梯度不稳定：每次更新只基于一个样本，梯度波动较大，可能导致优化过程不稳定。
可能停在局部最优：由于梯度具有随机性，可能在局部最优附近抖动，无法精确收敛。

六、最大似然估计(Maximum Likelihood Estimation, MLE，选读)

这个地方很难去考计算或者证明，但是属于你如果想学好这门课，必须要理解的一个理论
最小二乘回归的概率解释——最大似然估计 (这个地方会在第三周去详细解释)。

在讲 MLE 之前，我们先去讲线性回归的假设。

Linearity(线性): The relationship between x and the mean of y is linear.

$$y = x^T\theta + \varepsilon$$

Homoscedasticity(同方差性): The variance of residual is the same for any value of ε.

$$\mathrm{Var}(\varepsilon|x) = \sigma^2, \quad \forall x$$

Independence: Observations are independent of each other. 即误差项 ε 之间没有相关性

$$\mathbb{E}[\varepsilon_i \varepsilon_j] = 0, \quad i \neq j$$

Normality: For any fixed value of x, y is normally distributed

$$y|x \sim N(x^T\theta, \sigma^2)$$

这等价于假设误差项ε服从均值为 0，方差为σ2 的正态分布：

$$\varepsilon \sim N(0, \sigma^2)$$

MLE 的直观理解：利用已知的样本结果信息，反推最具有可能（最大概率）导致这些样本结果出现的模型参数值

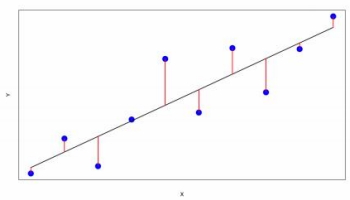
We can write the relationship between input variable x and output variable y as:

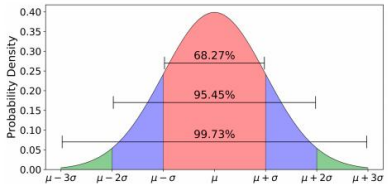
$$y_j = x_j^T\theta + \varepsilon_j$$

And ε_j is an error term which might be unmodeled effect or random noise. Let's assume ε_js are independent and identically distributed ($i.i.d.$) according to a Gaussian distribution:

$$\varepsilon_j \sim N(0, \sigma^2)$$

$$p(\varepsilon_j) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{\varepsilon_j^2}{2\sigma^2})$$





我们可以将输入变量 x 和输出变量 y 之间的关系表示为：

$$y_j = x_j^T\theta + \varepsilon_j$$

其中，ε_j 是一个**误差项**，它可能表示未建模的影响或随机噪声。我们假设所有的 ε_j 服从**独立同分布（i.i.d.）**，并且服从**高斯分布（正态分布）**：

$$\varepsilon_j \sim N(0, \sigma^2)$$

误差项的概率密度函数（PDF）为：

$$p(\varepsilon_j) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\varepsilon_j^2}{2\sigma^2}\right)$$

什么是独立同分布(i.i.d)?

独立（Independent）：每个随机变量之间相互独立，意味着一个变量的取值不会影响其他变量的取值。
同分布（Identically Distributed）：所有随机变量都服从相同的概率分布，即它们的分布规律相同，具有相同的均值、方差等统计特性。

slides 上的原文是：

这意味着：

$$p(\varepsilon_j) = p(y_j|x_j;\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_j - x_j^T\theta)^2}{2\sigma^2}\right)$$

因此，我们希望估计参数 θ，使得在所有 m 个训练样本的情况下，**输出 y 的概率最大**，即：

$$\mathcal{L}(\theta) = p(\mathbf{y}|X;\theta)$$

（这被称为**似然函数（Likelihood function）**）

为什么误差的概率等于输出 y 的概率?

我们之前假设**误差项 ε_j 服从正态分布**：

$$\varepsilon_j \sim N(0, \sigma^2)$$

由于 $y_j = x_j^T\theta + \varepsilon_j$，可以推导出 y_j 的条件分布：

$$y_j|x_j;\theta \sim N(x_j^T\theta, \sigma^2)$$

这意味着，在给定输入 x_j 时，输出 y_j 服从一个**均值为 $x_j^T\theta$，方差为 σ^2 的正态分布**。

正态分布的概率密度函数（PDF）为：

$$p(y_j|x_j;\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_j - x_j^T\theta)^2}{2\sigma^2}\right)$$

这说明，在参数 θ 给定的情况下，y_j 发生的概率由这个公式决定。

似然函数（Likelihood Function）表示的是在给定参数 θ 下，数据 y 发生的概率。假设数据点是独立同分布（i.i.d.）的，那么整个数据集的联合概率（即似然函数）是：

$$\mathcal{L}(\theta) = P(\mathbf{y}|X;\theta) = \prod_{j=1}^{m} p(y_j|x_j;\theta)$$

将上面的正态分布概率密度函数代入，我们得到：

$$\mathcal{L}(\theta) = \prod_{j=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_j - x_j^T\theta)^2}{2\sigma^2}\right)$$

这就是最大似然估计的目标函数，它衡量在给定参数 θ 时，整个训练数据集(X, y)发生的可能性。

我们的目标是找到最优的参数 θ，使得似然函数 L(θ) 最大化：

$$\hat{\theta} = \arg\max_{\theta} \mathcal{L}(\theta)$$

由于求乘积的优化比较复杂，我们通常取对数似然函数（log-likelihood function）来转换为求和形式：

$$\log \mathcal{L}(\theta) = \sum_{j=1}^{m} \left(-\frac{1}{2} \log(2\pi\sigma^2) - \frac{(y_j - x_j^T\theta)^2}{2\sigma^2} \right)$$

忽略与 θ 无关的常数部分，最大化对数似然等价于**最小化残差平方和（RSS）**：

$$\hat{\theta} = \arg\min_{\theta} \sum_{j=1}^{m} (y_j - x_j^T\theta)^2$$

这正是最小二乘法（OLS, Ordinary Least Squares）的目标函数！因此，在误差服从正态分布的情况下，OLS 就是 MLE 的特例。

MLE 和梯度下降的关系是什么？
MLE 提供了参数估计的理论框架，而梯度下降是用于求解 MLE 目标函数的优化方法之一。

七、Bias-Variance trade off

在机器学习中，我们用训练数据集去训练一个模型，通常的做法是定义一个误差函数，通过将这个误差的最小化过程，来提高模型的性能。然而我们学习一个模型的目的是为了解决训练数据集这个领域中的一般化问题，单纯地将训练数据集的损失最小化，并不能保证在解决更一般的问题时模型仍然是最优，甚至不能保证模型是可用的。这个训练数据集的损失与一般化的数据集的损失之间的差异就叫做**泛化误差**（generalization error）。

而泛化误差可以分解为**偏差（Biase）、方差（Variance）和噪声（Noise）**。

Bias 是用所有可能的训练数据集训练出的所有模型的输出的平均值与真实模型的输出值之间的差异。

$$bias^2(\boldsymbol{x}) = \left(\bar{f}(\boldsymbol{x}) - y\right)^2$$

偏差度量了学习算法的期望预测与真实结果的偏离程度，即刻画了学习算法本身的拟合能力。

Variance 是不同的训练数据集训练出的模型输出值之间的差异。

$$var(\boldsymbol{x}) = \mathbb{E}_D\left[\left(f(\boldsymbol{x}; D) - \bar{f}(\boldsymbol{x})\right)^2\right]$$

方差度量了同样大小的训练集的变动所导致的学习性能的变化，即刻画了数据扰动所造成的影响。

偏差度量的是单个模型的学习能力，而方差度量的是同一个模型在不同数据集上的稳定性。

噪声的存在是学习算法所无法解决的问题，数据的质量决定了学习的上限。假设在数据已经给定的情况下，此时上限已定，我们要做的就是尽可能的接近这个上限。

It can be shown that, when we assume $y = f + \varepsilon$ and we estimate f, with $\hat{f}$, then the expectation of error:

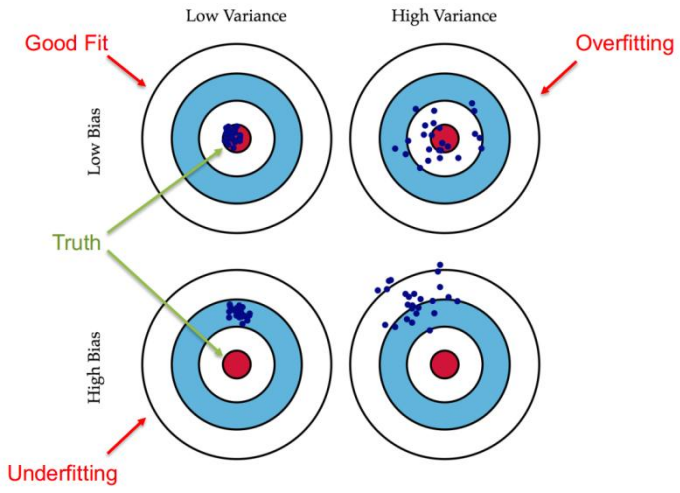
$$E[(y - \hat{f})^2] = (f - E[\hat{f}])^2 + Var(\hat{f}) + Var(\varepsilon)$$

So., the mean of squared error (MSE) can be written as:

$$MSE = Bias^2 + Variance + irreducable\ error$$

- **Irreducible error** or inherent uncertainty is associated with a natural variability in a system (noise). It can not be not reduced since it is due to unknown/unpredictable factors or simply due to chance.
- **Reducible error**, as the name suggests, can be and should be minimized further by adjustments to the model.

这表明误差可以分解为三部分：
Bias² 和 Variance 是可优化的，我们可以通过调整模型来降低这些误差。
不可约误差（Irreducible Error） 是由于噪声或不可预测的因素导致的，无法通过优化模型来减少。

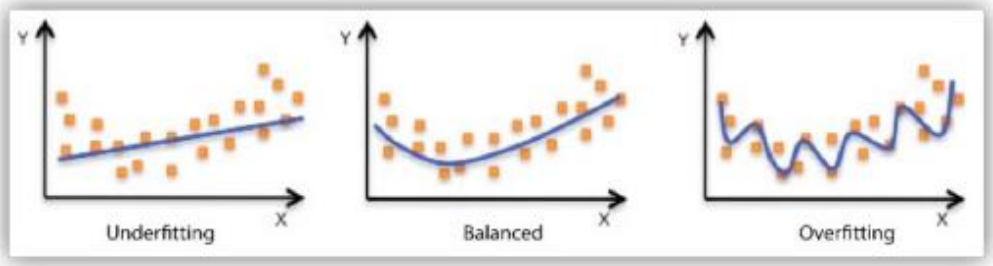


右上象限：低偏差 + 高方差（Overfitting ）

预测点（蓝色点）大多数围绕红色靶心，但分布非常分散。说明模型能很好地拟合训练数据**（低偏差）**，但泛化能力差**（高方差）**。这是过拟合（Overfitting）的典型情况，模型学习到了训练数据的细节和噪声，导致泛化能力差。适用于**训练集上表现很好，但在测试集上表现不佳的模型。**

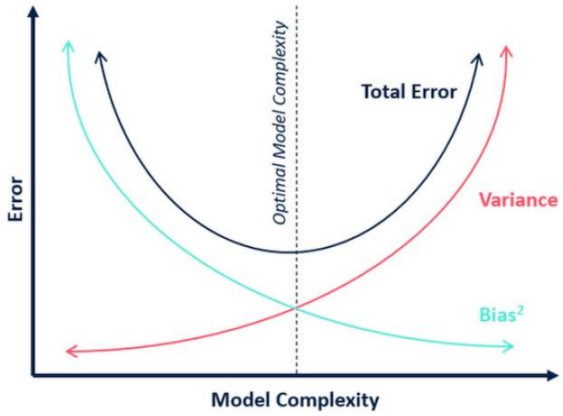
左下象限：高偏差 + 低方差（Underfitting ）

预测点（蓝色点）都集中在某个地方，但远离靶心。说明模型的预测结果稳定（低方差），但有系统性偏差（高偏差）。这是欠拟合（Underfitting）的情况，**模型过于简单，无法捕捉数据的真实模式。**适用于训练集和测试集上的表现都较差的模型。



一般来说，简单的模型会有一个较大的偏差和较小的方差，复杂的模型偏差较小方差较大。
欠拟合：模型不能适配训练样本，有一个很大的偏差。
过拟合：模型很好的适配训练样本，但在测试集上表现很糟，有一个很大的方差。

我们有什么办法尽量减少它对模型的影响呢?
一个好的办法就是正确选择模型的复杂度。复杂度高的模型通常对训练数据有很好的拟合能力，但是对测试数据就不一定了。而复杂度太低的模型又不能很好的拟合训练数据，更不能很好的拟合测试数据。因此，模型复杂度和模型偏差和方差具有如下图所示关系。



以及，正则化(Regularisation)。

七、正则化 Regularisation

什么是正则化?
正则化（Regularization）是一种防止过拟合（Overfitting）的方法，它通过对权重向量（weight vector）施加额外约束来实现。一种常见的做法是确保权重在平均意义上尽可能小，这种方法被称为收缩（Shrinkage）。

我们可以在损失函数中添加惩罚项（Penalty Term），使得回归系数（Regression Coefficients）收缩至接近零。 ——**收缩（Shrinkage）意味着我们对模型的参数施加约束，使其不会变得过大，从而降低模型的复杂度。**

岭回归(Ridge Regression)

假设均方误差（MSE, Mean Squared Error）是损失函数，则正则化后的损失函数可以表示为：

$$J(\theta) = \sum_{j=1}^{m}(y_j - h_\theta(x_j))^2 + \lambda \sum_{i=1}^{n} \theta_i^2$$

其中，第一项是标准的 MSE 损失，用于衡量模型预测值与真实值的误差。
第二项是**正则项（Regularization Term）**，用于控制参数 θ 的大小。

λ 是正则化强度的超参数，控制正则项的权重：

- λ 过大：模型过于简单，可能欠拟合。
- λ 过小：模型接近普通最小二乘（OLS），可能过拟合。

最小二乘回归（Least-Square Regression）本质上是一个优化问题，其优化目标为：

$$\theta^* = \arg\min_\theta (y - X\theta)^T (y - X\theta)$$

正则化后的版本（岭回归 Ridge Regression）如下：

$$\theta^* = \arg\min_\theta (y - X\theta)^T (y - X\theta) + \lambda ||\theta||^2$$

正则化问题仍然可以通过封闭解（Closed-form Solution）求解：

$$\theta = (X^T X + \lambda I)^{-1} X^T y$$

正则化相当于在矩阵 $X^T X$ 的对角线上加上 λ，这是一种常见的数值计算方法，可以提高矩阵求逆的数值稳定性。

此外，还有另一种常见的正则化方法：

LASSO 回归（Least Absolute Shrinkage and Selection Operator）：LASSO 用 $\sum_i |\theta_i|$ 代替 $\sum_i \theta_i^2$

即使用 L1 范数进行正则化，而不是 L2 范数。
结果：LASSO 不仅能收缩权重，还能将某些权重设为 0，从而进行**特征选择（Feature Selection）**。
LASSO 适用于高维数据，能够自动筛选出最重要的变量，得到稀疏解（Sparse Solution）。

八、Exercise:

我们有以下简单的数据集，只有一个特征 x 和一个目标变量 y:

x y
1 2
2 2.8
3 3.6

假设模型是一个简单的线性回归模型:

$$h_\theta(x) = \theta_0 + \theta_1 x$$

取学习率η=0.1，θ0=0，θ1=0

我们使用梯度下降法来找到最优参数 θ0 和 θ1，使得模型预测值与真实值之间的均方误差 (MSE) 最小。

算法过程:

1. 初始化θ_0 向量的值
2. 计算损失函数的梯度

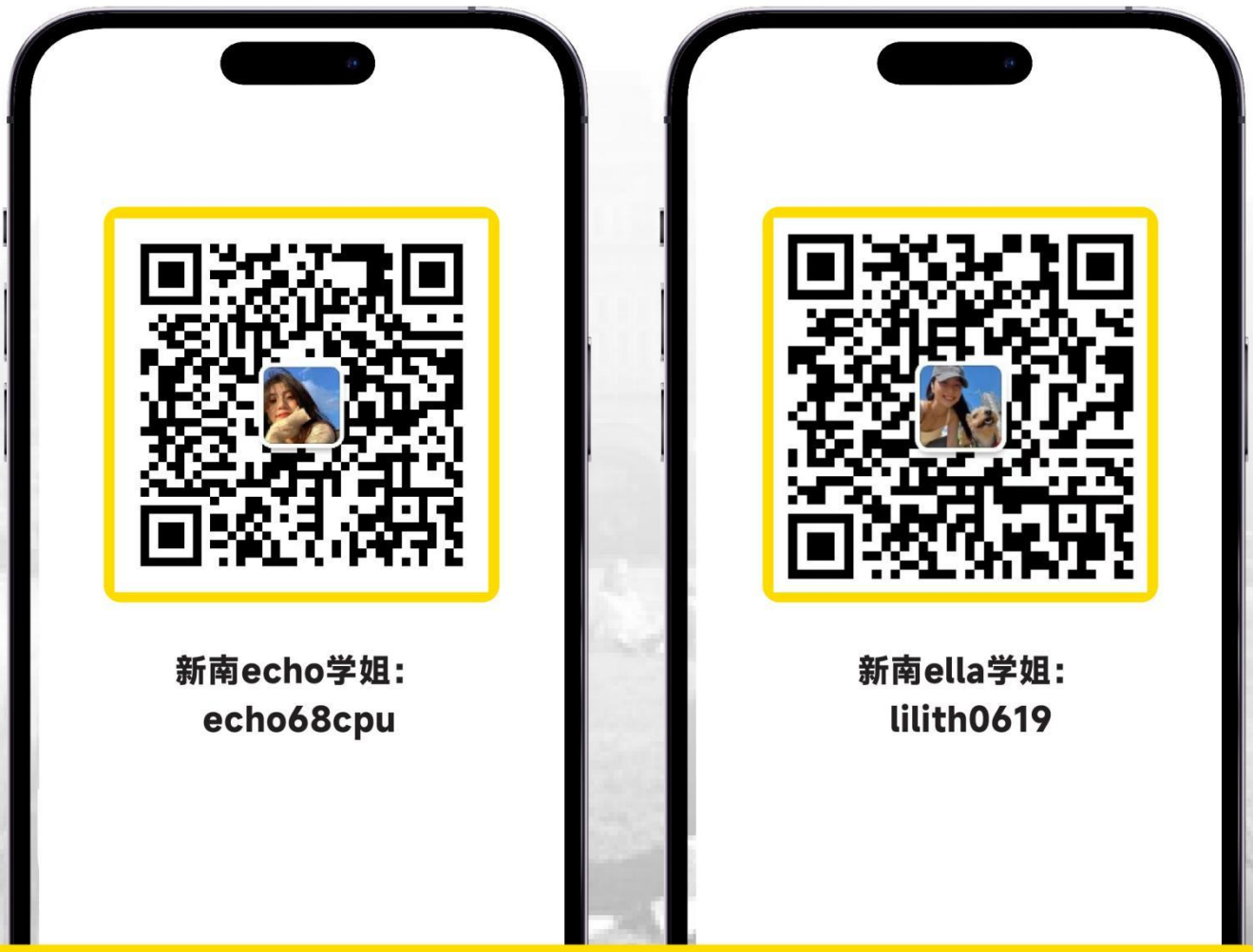
$$\nabla_\theta J(\theta) = \frac{1}{2m} \cdot 2X^T(X\theta - Y)$$

化简得到,

$$\nabla_\theta J(\theta) = \frac{1}{m} \cdot X^T(X\theta - Y)$$

3. 将其代入 $\nabla_\theta J(\theta)$ 得到当前位置的梯度$\nabla_\theta J(\theta_0)$;
4. 更新参数$\theta_1 = \theta_0 - \alpha\nabla_\theta J(\theta_0)$
5. 重复以上步骤，直到更新到某个 θ_k，达到停止条件，这个 θ_k就是我们求解的参数向量。

CPU EDUCATION



CPU·EDU成长有你陪伴

·让海外学习更轻松·