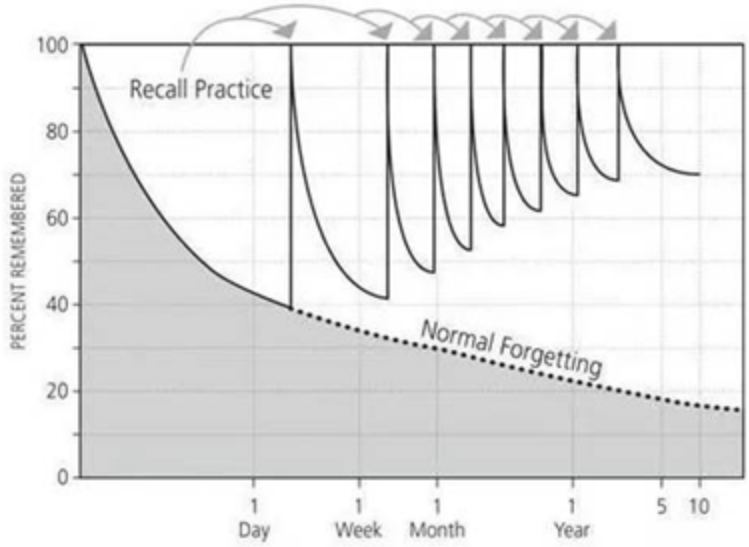
关于复习算法的设计

本文档用于记录对于复习功能的算法设计思考，最终成品采用的算法可能与之不同。

当前的设想为使用个性化优化的 Supermono-2 算法，我们可以将其称之为 **Optimized SM-2**。

算法选型

一个经典的记忆模型就是对数 Ebbinghaus 遗忘曲线。但是该曲线在实际应用场景中严重受阻，例如在 20 分钟后，记忆量就会下降到 58.2%，我们不可能要求用户每隔几十分钟就复习一次之前的题目，这样会极大的降低用户的有效学习量。



我们需要一个可应用的记忆算法模型。因此，采用设计简单、开源的算法 Supermemo-2 并进行优化。设 EF 表示条目的容易程度，I 表示复习的间隔（单位：天），q 表示用户对本次复习质量的评分。根据用户本次的复习质量更新下一次复习的时间间隔：

$$I_{\text{new}}(I, \text{EF}, q) = \begin{cases} 1 & q < 3 \\ \max(1, \lfloor I \times \text{EF} \rfloor) & q \geq 3 \end{cases}$$

其中 q 的评分标准为：

评分	用户表现
5	完美回忆
4	较难但正确回忆

评分	用户表现
3	困难但最终正确回忆
2	错误回忆（但看到答案后能理解）
1	完全遗忘
0	完全不会

最后更新易度因子 EF：

$$EF' = EF + (0.1 - (5 - q) \times (0.08 + (5 - q) \times 0.02)) \qquad (*)$$

我们关键需要解决的问题是：

1. 在每次用户答题之后，自动评分，即自动生成 q 值；
2. 在用户初次答题后，不使用传统 EF 计算公式生成 EF ，因为这样会导致刚开始的 EF 没有明显的分层。为满足用户额的快速复习需求，采用通过对于用户特征的自学习，更好的根据初次答题的参数调整 EF 值。

评分标准的测定

传统使用 SM-2 及类似衍生算法的软件常常在复习完成一道题后要求用户给出 q 分数。虽然可行，但是在实际测试中，这通常会明显的降低用户体验。

因此，我们使用答题时间 t、最近三次的质量评分平均数 $\bar{q}$、以及用户答案与标准答案的 Levenshtein 距离 L 定量的衡量用户本次答题的 q 值。

形式化地说，给定特征向量 $\mathbf{x} = (t, \bar{q}, L)$，预测质量评分 $q \in \{0, 1, 2, 3, 4, 5\}$。

假设上述特征互相独立，选用朴素贝叶斯模型：

$$P(q = k \mid t, \bar{q}, L) = \frac{P(q = k) \cdot P(t \mid q = k) \cdot P(\bar{q} \mid q = k) \cdot P(L \mid q = k)}{\sum_{j=0}^{5} P(q = j) \cdot P(t \mid q = j) \cdot P(\bar{q} \mid q = j) \cdot P(L \mid q = j)}$$

交互特征处理

对于答题时间而言，考虑到答案长短不均一，我们使用输入速率 r 来评价答题时间指标。该指标可以广泛的反应用户输入速度快慢，删改文字等情况。该指标与 Levenshtein 距离存在强交互作用，一个典型的示例如下：

输入速率 r	相似度 S	可能情况
高 (>3)	高 (>0.9)	熟练回忆（$q \geq 4$）
高 (>3)	低 (<0.5)	瞎jb猜 ($q \leq 1$)
低 (<1)	高 (>0.9)	努力回忆但最终正确（$q \geq 3$）
低 (<1)	低 (<0.5)	努力回忆但失败 ($q \leq 2$)

其中相似度 S 是 Levenshtein 距离 L 的归一化相似度。因此我们构建组合特征 $F = f(r, L)$ ，显然此时 F 与 $\bar{q}$ 相互独立，朴素贝叶斯输入为 $[F, \bar{q}]$，表达式为：

$$P(q = k \mid F = f, H = h) = \frac{P(q = k) \cdot P(F = f \mid q = k) \cdot P(H = h \mid q = k)}{\sum_{j=0}^{5} P(q = j) \cdot P(F = f \mid q = j) \cdot P(H = h \mid q = j)}$$

$$H = \text{discretize}(\bar{q}) \in \{\text{poor}, \text{medium}, \text{good}\}$$

数据归一化

Levenshtein 距离 L（编辑距离）定义为：将字符串 T 转换为字符串 R 所需的最少单字符编辑操作次数，允许的操作包括：

- 插入（Insertion）
- 删除（Deletion）
- 替换（Substitution）

归一化 Levenshtein 相似度 S 定义为：

$$S(A, B) = 1 - \frac{L(A, B)}{\max(|A|, |B|)}$$

相似度结果 $S \in [0, 1]$，其中 $S = 1$ 表示完全匹配（包括空字符串匹配空字符串），而 $S = 0$ 表示完全不相似（一个字符串是另一个字符串的最大编辑距离）

数据离散化

朴素贝叶斯是概率模型，适用于处理离散化特征，因此我们构造离散化函数，将以上指标离散化。

对于 组合特征 F 而言，我们首先离散化组成 F 的特征 r 和 L 。用于用户的打字速度可能存在差异，我们提前测量用户的目视打字速度 A，计算用户的相对答题速度：

$$r_{rel} = \frac{r}{\mathrm{A}}$$

定义离散化函数 speed_level 为：

$$\text{speed\_level}(r_{\text{rel}}) = \begin{cases} \text{very\_fast} & r_{\text{rel}} > 0.8 \\ \text{fast} & 0.6 < r_{\text{rel}} \leq 0.8 \\ \text{normal} & 0.4 < r_{\text{rel}} \leq 0.6 \\ \text{slow} & 0.2 < r_{\text{rel}} \leq 0.4 \\ \text{very\_slow} & r_{\text{rel}} \leq 0.2 \end{cases}$$

由 r_{rel} 和 L 组合为 F，由于情况较多，我们使用表格的方式将答题行为映射为离散的认知模式标签 `F` ，输入为用户**相对输入速率 (r_rel)** 和 **Levenshtein相似度 (S)**。

r_{rel} 区间	S 区间	组合特征 F (认知模式)	典型解释与预期q值关联
r_rel > 0.8	**S ≥ 0.95**	`fast_exact`	**熟练回忆**：近乎本能地快速准确回答。关联高q值（4或5）。
r_rel > 0.8	**S < 0.50**	`fast_wrong`	**快速猜测**：未加思考快速作答且错误。关联低q值（0或1）。
0.4 < r_rel ≤ 0.8	**S ≥ 0.95**	`normal_exact`	**正常回忆**：速度正常，结果准确。关联中高q值（3或4）。
0.4 < r_rel ≤ 0.8	**0.50 ≤ S < 0.80**	`normal_partial`	**部分记忆**：速度正常但答案有瑕疵。关联中等q值（2或3）。
0.4 < r_rel ≤ 0.8	**S < 0.50**	`normal_wrong`	**正常错误**：速度正常但答案错误。关联低q值（1或2）。
r_rel ≤ 0.4	**S ≥ 0.80**	`slow_correct`	**困难回忆**：经过努力和犹豫后答对。关联中等q值（3或4）。
r_rel ≤ 0.4	**S < 0.50**	`slow_wrong`	**困难失败**：努力尝试后仍然答错。关联低q值（0或1）。

对于历史 q 值情况而言，首先求得历史三次的平均 $\bar{q}$ 值：

$$\bar{q} = \frac{1}{3}\sum_{i=1}^{3} q_{t-i}$$

之后采用等距分箱方案对 $\bar{q}$ 离散化：

$$H(\bar{q}) = \begin{cases} \text{poor} & \bar{q} < 2.0 \\ \text{medium} & 2.0 \le \bar{q} < 3.5 \\ \text{good} & \bar{q} \ge 3.5 \end{cases}$$

初始易度因子的测定

初始易度因子 EF_0 可以由用户的作答行为映射，由于 EF 是连续型变量，无需对 S 和 r_{rel} 进行离散化。

由于并没有充分的证据的说明 EF 与 S 和 r_{rel} 之间存在的关系，因此考虑使用梯度提升树模型 FastTree 预测 EF_0 的值。

对于包含 M 棵回归树的集成模型，其最终预测函数 F 定义为各树输出值的加权和：

$$EF_{pred} = F_M(S, r_{rel}) = F_0 + \eta * \Sigma_{m=1}^{M} h_m(S, r_{rel})$$

每棵树 h_m 的构建旨在最小化一个可微的损失函数 Loss ，当前计划使用均方误差 MSE，即：

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

算法采用前向分步策略。

自学习方式

在第一次生成初始 EF 之后，如前文所述，下一次的 EF 值可以使用 $(*)$ 式计算。记录用户在后续答题过程中的相似度 S，相对输入速度 r_{rel}，以及通过 $(*)$ 式计算的 EF 值用于训练模型即可。

模型训练耗时可能较长，可以使用 `BackgroundWorker` 等组件创建一个后台进/线程，通过读取用户的电脑配置，尽可能的在后台进行无感化训练，为用户生成一个独一无二的，真正匹配用户行为特征的模型。

后记

作为一个做工程为主，较少涉及到算法的开发者而言，这个文档里记录的方法只是一个 “俺寻思能行” 的方案，并未得到严格的数学证明，开发者本人页没有足够的资金和精力展开大规模的实验。欢迎有能力的算法工作人员帮助修正和优化相关算法。

无感化AI一直是我近期在思考的主题，很多集成AI技术的应用事实上只是提供了一个对话框，然后把相关的内容和用户请求一并发送给大模型。相对于“聊天机器人”，我对大模型的认知更偏向于“一种数据处

理的方法”，它可以在处理一些我们之前感到望尘莫及的数据，例如从复杂的文本内容中提取复习题目。这个项目也是我对无感化AI的一次尝试、一次探索。

春风有信，花开有期。愿华枝春满，岁月安暖。

参考文献

[1] Yujian, L., & Bo, L. (2007). A normalized Levenshtein distance metric. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6), 1091–1095. https://doi.org

[2] Jankowski, J. (2022, November 2). Application of a computer to improve the results obtained in working with the SuperMemo method - SuperMemo. SuperMemo. https://www.supermemo.com/en/blog/application-of-a-computer-to-improve-the-results-obtained-in-working-with-the-supermemo-method