

# 《推荐系统》实验报告

学号：20002462      姓名：刘子言      班级：计 203      成绩：

实验名称：CRM 系统之推荐算法	实验地点：信息楼 211																								
所使用的工具软件及环境：Pycharm（Python3.8），SqlServer																									
<b>一、实验目的：</b> 熟悉关联算法、相邻方法，能应用关联算法或相邻方法设计个性化推荐系统，如推荐相应的优惠折扣或用户可能感兴趣的内容																									
<b>二、实验内容：</b> 1、输入客户数据 2、应用关联算法或相邻方法等计算该用户适宜的优惠政策或可能感兴趣的内容 3、输出相应的推荐内容																									
<b>三、实验步骤：</b> <b>1、请简要叙述系统应用技术（如应用的编程语言、数据库或文件技术等）、推荐算法</b> (1) 编程语言：Python 语言，界面利用 GUI 实现，系统运用 Pycharm 软件开发； (2) 数据库：SqlServer 数据库； (3) 推荐算法：关联算法。  <b>2、请叙述数据存储的方式（如果是数据库，请列出表结构；如果是文件，请列出文件中记录的结构），并说明数据的来源（若是自动生成的，介绍一下生成规则）</b> <b>(1) 本系统采用 SqlServer 数据库存储数据</b> 数据库名称：CRM_db 共设计 3 张原始数据的数据表，每张表的表结构如下所示：  • 用户表（表名：User1） <table border="1"><thead><tr><th>Column</th><th>Type</th><th>Nullable</th><th>Indexes</th></tr></thead><tbody><tr><td><u>userId</u></td><td>nchar(6)</td><td>no</td><td>primary</td></tr><tr><td>userName</td><td>nvarchar(20)</td><td>no</td><td></td></tr></tbody></table>  • 商品表（表名：Goods） <table border="1"><thead><tr><th>Column</th><th>Type</th><th>Nullable</th><th>Indexes</th></tr></thead><tbody><tr><td><u>goodsId</u></td><td>nchar(6)</td><td>no</td><td>primary</td></tr><tr><td>goodsName</td><td>nvarchar(20)</td><td>no</td><td></td></tr></tbody></table>		Column	Type	Nullable	Indexes	<u>userId</u>	nchar(6)	no	primary	userName	nvarchar(20)	no		Column	Type	Nullable	Indexes	<u>goodsId</u>	nchar(6)	no	primary	goodsName	nvarchar(20)	no	
Column	Type	Nullable	Indexes																						
<u>userId</u>	nchar(6)	no	primary																						
userName	nvarchar(20)	no																							
Column	Type	Nullable	Indexes																						
<u>goodsId</u>	nchar(6)	no	primary																						
goodsName	nvarchar(20)	no																							

price	float	no	
skinType	nvarchar(10)	no	

• 购买记录表（表名：Buy）

Column	Type	Nullable	Indexes
<u>userId</u>	nchar(6)	no	primary
<u>goodsId</u>	nchar(6)	no	primary

## （2）数据来源

本系统数据库中的数据来源于本组成员通过查阅相关资料编写生成的数据。其中：

用户表（User1）中的数据有 24 条，代表 24 个独立的用户。本系统共涉及 6 种肤质（干皮、混干、混油、油皮、干敏、油敏），平均每种肤质有 4 位用户；

商品表（Goods）中的数据有 30 条，代表本平台售卖的 30 种护肤品。本系统共涉及 6 种肤质，小组成员通过查阅资料，为每种肤质均挑选了 5 种适合的具有代表性的商品，并记录其名称、价格以及图片；

购买记录表（Buy）中的数据有 60 条，代表 24 位用户曾在本平台购买的产品的历史记录。本系统共涉及 6 种肤质，小组成员根据各个护肤品之间的价格、互补功效以及品牌效应，编写了每位用户可能多次或同时购买的多个商品记录，具有一定的参考价值与真实性；其中平均每种肤质有 10 条相关商品的购买记录。

## 3、请叙述系统的功能模块和具体的实现步骤（请画出所应用算法的流程图）

本系统的功能模块主要分为两大部分：一是展现用户想要购买的产品的详细信息，二是推荐适合用户肤质的护肤品并展示其相关信息。

### （1）用户输入输出设计

用户需要在两个下拉框中分别选择自己的肤质，以及想要购买的商品名称；输出采用 GUI 设计界面，传入参数与后端进行交互。

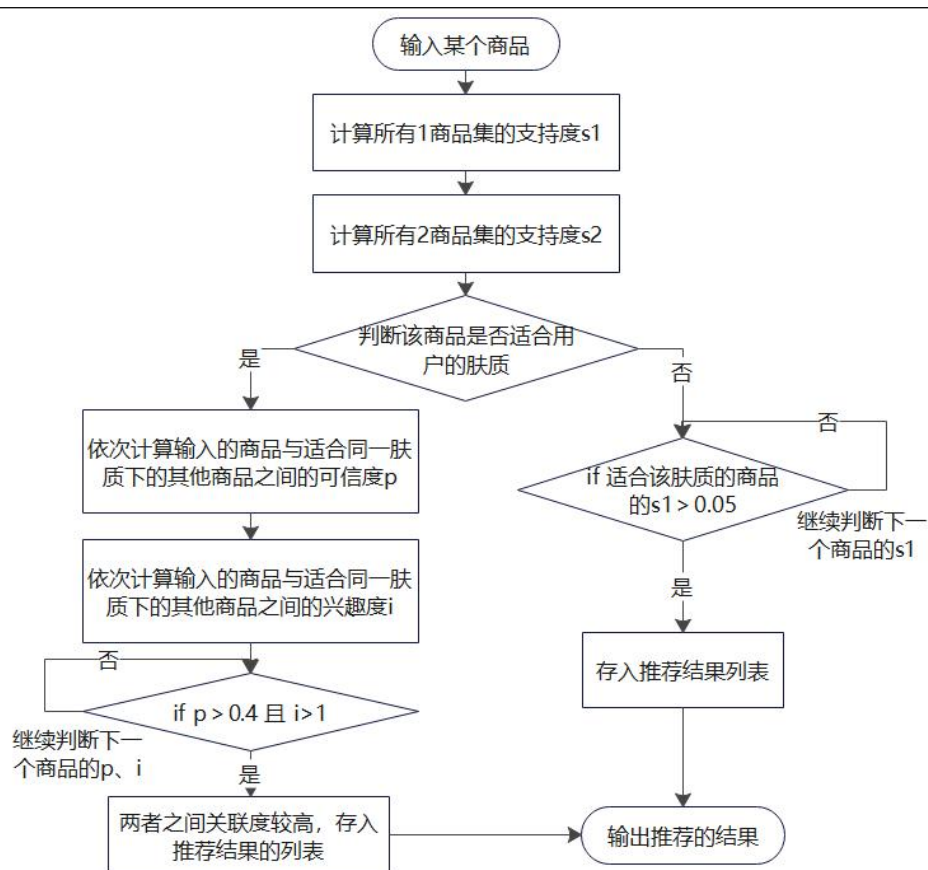
### （2）原商品信息的展示

主要实现思路：记录用户在前端输入的参数（包括肤质以及想要购买的商品名称），根据输入的商品名称找到数据库 Goods 表中对应的商品的序号，然后根据该商品序号在前端返回商品的名称、价格、适合肤质以及商品图片。

### （3）推荐商品信息的展示

#### • 关联算法

本系统中运用的关联算法的流程图如下：



注：在本推荐系统的应用中，频繁项集只涉及两项。

### • 实现思路

将用户想要购买的商品所适合的肤质信息与用户选择的肤质信息进行比较，如果相同，则推荐与该商品相关联的商品；如果不相同，则返回提示语句，并推荐适合该肤质的商品中支持度符合要求的那一部分商品。

其中，可信度的临界值设为 0.4，兴趣度为 1，最小支持度设为 0.05。

#### • 支持度计算公式：

$$s(A) = \text{count}(A) / N \quad \text{OR} \quad s(AB) = \text{count}(AB) / N$$

在计算单个商品集的支持度时运用单层循环，计算 2 个商品的集合的支持度时运用了双层循环嵌套。

#### • 可信度计算公式：

$$p = s(AB) / s(A)$$

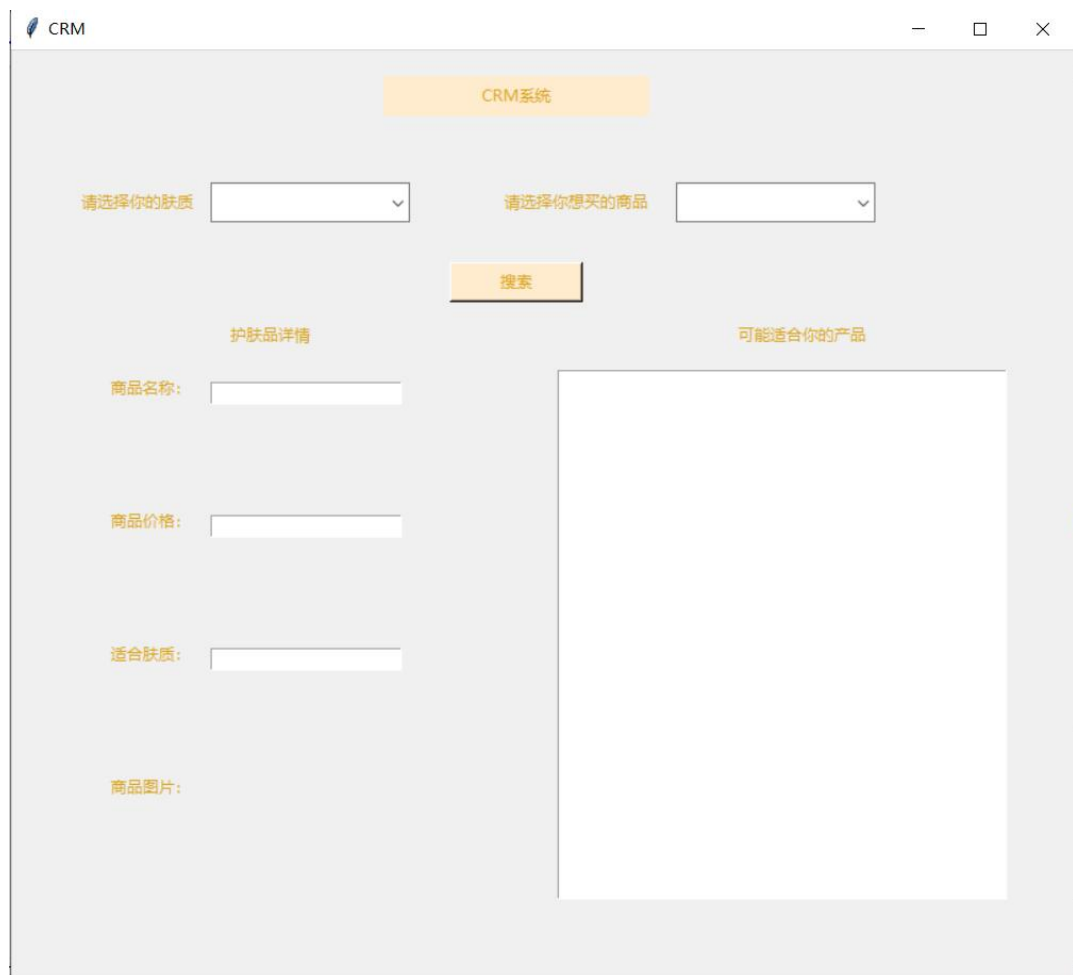
#### • 兴趣度计算公式：

$$i = s(AB) / s(A) * s(B)$$

## 四、系统运行情况界面

### 1、初始化界面

界面上侧是用户需要输入的信息，包括肤质和商品名称；下册是搜索结果，包括所选商品详情以及平台推荐结果。



The image shows a web browser window titled "CRM" with a standard window control bar. The main content area has a light gray background. At the top center, there is a yellow rectangular button labeled "CRM系统". Below this, there are two dropdown menus. The first is labeled "请选择你的肤质" (Please select your skin type) and the second is labeled "请选择你想买的商品" (Please select the product you want to buy). Both dropdown menus have a small downward arrow icon. Below the dropdown menus is a yellow rectangular button labeled "搜索" (Search). Below the search button, the interface is divided into two columns. The left column is titled "护肤品详情" (Skincare Product Details) and contains four input fields: "商品名称:" (Product Name:), "商品价格:" (Product Price:), "适合肤质:" (Suitable Skin Type:), and "商品图片:" (Product Image:). The right column is titled "可能适合你的产品" (Products that may suit you) and contains a large, empty white rectangular area.

### 2、用户选择自己的肤质，以及想要购买的护肤品的名称



The image shows the same CRM system interface as above, but with the "请选择你的肤质" dropdown menu open. The dropdown menu displays a list of skin types: "干皮" (Dry skin), "混干皮" (Combination dry skin), "油皮" (Oily skin), "混油皮" (Combination oily skin), "干敏皮" (Dry sensitive skin), and "油敏皮" (Oily sensitive skin). The "干皮" option is currently selected and highlighted in blue. The "搜索" button and the "可能适合你的产品" section remain visible and unchanged.



3、点击搜索按钮，推荐系统在下方展示推荐结果



(1) 展示区左侧

展示用户想要购买的护肤品详情，包括护肤品名称、价格、适合肤质以及商品的图片。

## (2) 展示区右侧

展示推荐系统推荐的可能适合用户肤质的商品信息。

- 如果用户想要购买的产品适合其肤质，则根据关联算法推荐除该产品外其他可能适合该用户的护肤品，例如：

CRM

CRM系统

请选择你的肤质

混干皮

请选择你想买的商品

934神经酰胺酵母水乳

搜索

护肤品详情

商品名称:

934神经酰胺酵母水乳

商品价格:

188.0

适合肤质:

混干皮

商品图片:

可能适合你的产品

000007 初夏堂山茶花水乳, 199.0 混干皮  
000009 瑛尔博士益生菌水乳, 148.0 混干皮

## CRM系统

请选择你的肤质

混油皮



请选择你想买的商品

溪木源山茶花水乳



搜索

## 护肤品详情

商品名称:

溪木源山茶花水乳

商品价格:

259.0

适合肤质:

混油皮

商品图片:



## 可能适合你的产品

000014 初夏堂金缕梅水乳 218.0 混油皮  
000015 934神经酰胺控油水乳 228.0 混油皮

## CRM系统

请选择你的肤质

油皮



请选择你想买的商品

理肤泉b5面霜40ml



搜索

## 护肤品详情

商品名称:

理肤泉b5面霜40ml

商品价格:

119.0

适合肤质:

油皮

商品图片:



## 可能适合你的产品

000017 PMPM海糖进阶版水乳 189.0 油皮  
000018 tfit深层滋养水乳 109.0 油皮



- 如果用户选择的产品不适合其肤质，则给出文字提示，并根据关联算法推荐适合这一肤质的产品中支持度符合一定条件的护肤品，例如：

CRM

CRM系统

请选择你的肤质

干皮

请选择你想买的商品

黛珂牛油果水乳

搜索

护肤品详情

可能适合你的产品

商品名称:

黛珂牛油果水乳

商品价格:

268.8

适合肤质:

油敏皮

商品图片:



你想买的产品不适合你的肤质哦！  
我们为您推荐了以下适合您的护肤品：

000001 谷雨仙人掌水乳 149.0 干皮

000002 PMPM玫瑰红茶水乳 169.0 干皮

000003 珀莱雅水动力水乳 179.0 干皮

000004 肌研浓润型水乳 149.0 干皮

000005 自然堂雪域精粹水乳 219.0 干皮

## CRM系统

请选择你的肤质

干敏皮

请选择你想买的商品

自然堂雪域精粹水乳

搜索

## 护肤品详情

商品名称:

自然堂雪域精粹水乳

商品价格:

219.0

适合肤质:

干皮

商品图片:



## 可能适合你的产品

你想买的产品不适合你的肤质哦!

我们为您推荐了以下适合您的护肤品:

000027 PMPM干叶玫瑰水乳, 219.0 干敏皮

000029 韩束红胶囊水乳, 279.0 干敏皮

000030 科颜氏高保湿面霜50ml 330.0 干敏皮

## CRM系统

请选择你的肤质

油敏皮



请选择你想买的商品

肌研浓润型水乳



搜索

## 护肤品详情

商品名称:

肌研浓润型水乳

商品价格:

149.0

适合肤质:

干皮

商品图片:



## 可能适合你的产品

你想买的产品不适合你的肤质哦!

我们为您推荐了以下适合您的护肤品:

000023 欧倩龙胆敏感肌水乳, 229.0 油敏皮

000024 黛珂牛油果水乳, 268.8 油敏皮

000025 薇诺娜舒敏保湿水乳, 268.0 油敏皮

## 五、程序设计代码

### 1、前端 GUI 界面

# GUI 窗口基本配置

```
root = tk.Tk()
```

```
root.wm_title("CRM")
```

```
sw = root.winfo_screenwidth()
```

```
sh = root.winfo_screenheight()
```

```
ww = 800
```

```
wh = 700
```

```
x = (sw - ww) / 2
```

```
y = (sh - wh) / 2
```

```
root.geometry("%dx%d+%d+%d" % (ww, wh, x, y))
```

# 标题

```
label1 = tk.Label(root, text="CRM 系统", fg="#DAA520", bg="#FFEB CD")
```

# 用 label.pack()会使该布局无效

```
label1.place(width=200, height=30, x=280, y=20)
```

# 选择肤质

```
label2 = tk.Label(root, text="请选择你的肤质", fg="#DAA520")
```

```
label2.place(x=20, y=100, width=150, height=30)
```

# 肤质

```
type_of_skin = ['干皮', '混干皮', '油皮', '混油皮', '干敏皮', '油敏皮']
```

```
the_product_you_want = tk.ttk.Combobox(root, value=type_of_skin)
```

```
# the_product_you_want.pack(padx=5, pady=10)
```

```
the_product_you_want.place(x=150, y=100, width=150, height=30)
```

# 商品选择

```
label3 = tk.Label(root, text="请选择你想买的商品", fg="#DAA520")
```

```
label3.place(x=350, y=100, width=150, height=30)
```

# 商品

```
the_order_you_want = tk.ttk.Combobox(root, value=all_Goods_name)
```

```
the_order_you_want.place(x=500, y=100, width=150, height=30)
```

# “搜索”按钮

```
b1 = tk.Button(root, text="搜索", command=Good_detail, activebackground="#2082AA",  
fg="#DAA520", bg="#FFEB CD")
```

```
b1.place(x=330, y=160, width=100, height=30)
```

# 选择的商品详情

```
label5 = tk.Label(root, text="护肤品详情", fg="#DAA520")
```

```
label5.place(x=120, y=200, width=150, height=30)

# 显示商品详情界面
Label7 = tk.Label(root, text="商品名称: ", fg="#DAA520")
Label7.place(x=30, y=240, width=150, height=30)
show_product_name = tk.Entry(root, width=20, font=('宋体', '10'))
show_product_name.place(x=150, y=250)

Label8 = tk.Label(root, text="商品价格: ", fg="#DAA520")
Label8.place(x=30, y=340, width=150, height=30)
show_product_price = tk.Entry(root, width=20, font=('宋体', '10'))
show_product_price.place(x=150, y=350)

Label9 = tk.Label(root, text="适合肤质: ", fg="#DAA520")
Label9.place(x=30, y=440, width=150, height=30)
show_product_skin = tk.Entry(root, width=20, font=('宋体', '10'))
show_product_skin.place(x=150, y=450)

Label10 = tk.Label(root, text="商品图片: ", fg="#DAA520")
Label10.place(x=30, y=540, width=150, height=30)

# 推荐的商品
label6 = tk.Label(root, text="可能适合你的产品", fg="#DAA520")
label6.place(x=520, y=200, width=150, height=30)

# 推荐的商品详情界面
show_listbox = tk.Listbox(root)
show_listbox.place(x=410, y=240, width=340, height=400)

root.mainloop()
```

## 2、连接 CRM\_db 数据库及获取相关数据

```
# 连接数据表，调用数据分别存储在列表 User、Goods、Buy 中
conn = pymysql.connect(host='localhost', user='sa', password='lzy123456', database='CRM_db',
                        charset='UTF8')
cursor = conn.cursor()
# 读取各表信息存入列表
sql_select1 = "SELECT * FROM User1"
cursor.execute(sql_select1)
User = cursor.fetchall()

sql_select2 = "SELECT * FROM Goods"
cursor.execute(sql_select2)
Goods = cursor.fetchall()
```

```
sql_select3 = "SELECT * FROM Buy"
cursor.execute(sql_select3)
Buy = cursor.fetchall()

sql_select4 = "SELECT goodsName FROM Goods"
cursor.execute(sql_select4)
all_Goods_name = cursor.fetchall()

conn.commit()
conn.close()
```

```
# N 为用户总数
N = len(User)
```

### 3、后端操作与算法的实现

#### (1) 显示用户想要购买的商品详情

```
product_number = the_order_you_want.current()
skin_type_number = the_product_you_want.current()
skin_type = type_of_skin[skin_type_number]
show_listbox.delete(0, tk.END)
global item_number
global out_product_name
global price
global out_skin_type
global photo
item_number = Goods[product_number][0]
out_product_name = Goods[product_number][1]
price = Goods[product_number][2]
out_skin_type = Goods[product_number][3]
img_path = 'E:\《数学+计算机》\电商金融\小组项目\CRM\img\'+ item_number + '.jpg'
img = Image.open(img_path).resize((100, 120))
photo = ImageTk.PhotoImage(img)

show_product_name.delete(0, tk.END)
show_product_name.insert(0, out_product_name)
show_product_price.delete(0, tk.END)
show_product_price.insert(0, price)
show_product_skin.delete(0, tk.END)
show_product_skin.insert(0, out_skin_type)
# 显示商品详情界面
show_product_photo = tk.Label(root, image=photo)
show_product_photo.place(x=150, y=500, width=100, height=120)
# 判断用户想要购买的商品是否适合他的肤质
```

```
if skin_type == out_skin_type:
    recommend(out_product_name, skin_type, 1)
else:
    recommend(out_product_name, skin_type, 0)
```

## (2) 推荐算法（关联算法）的实现

# 单个商品的支持度计算

```
def support1(item_number):
    Buy_n = len(Buy)
    i = 0
    count = 0
    while i < Buy_n:
        if Buy[i][1] == item_number:
            count = count + 1
        i = i + 1
    support = count / N
    return support
```

# 两个商品的支持度计算

```
def support2(item_number1, item_number2):
    Buy_n = len(Buy)
    i = 0
    count = 0
    while i < Buy_n:
        j = 0
        if Buy[i][1] == item_number1:
            while j < Buy_n:
                if Buy[j][1] == item_number2 and Buy[j][0] == Buy[i][0]:
                    break
                j = j + 1
            if j < Buy_n:
                count = count + 1
        i = i + 1
    support = count / N
    return support
```

# 可信度计算

```
def confidence_level(item_number1, item_number2):
    n1_support = support1(item_number1)
    n2_support = support1(item_number2)
    n12_support = support2(item_number1, item_number2)
    if n2_support > min_support:
        reliability = n12_support / n1_support
    else:
```

```

        reliability = 0
    return reliability

# 兴趣度计算
def interestingness(item_number1, item_number2):
    n1_support = support1(item_number1)
    n2_support = support1(item_number2)
    n12_support = support2(item_number1, item_number2)
    if n2_support > min_support:
        interest = n12_support / (n1_support * n2_support)
    else:
        interest = 0
    return interest

# 推荐总函数
def recommend(product_name, skin_type, flag):
    global recommend_result
    recommend_result = []
    if flag == 1:
        i = 0
        Goods_n = len(Goods)
        while i < Goods_n:
            if Goods[i][3] == skin_type and Goods[i][1] != product_name:
                reliability = confidence_level(item_number, Goods[i][0])
                interest = interestingness(item_number, Goods[i][0])
                if reliability > min_reliability and interest > min_interest:
                    recommend_result.append(Goods[i])
            i = i + 1
        elif flag == 0:
            notice = '你想买的产品不适合你的肤质哦！'
            show_listbox.insert(tk.END, notice)
            notice = '我们为您推荐了以下适合您的护肤品：'
            show_listbox.insert(tk.END, notice)
            i = 0
            Goods_n = len(Goods)
            while i < Goods_n:
                if Goods[i][3] == skin_type:
                    if support1(Goods[i][0]) > min_support:
                        recommend_result.append(Goods[i])
                i = i + 1

    for i in recommend_result:
        show_listbox.insert(tk.END, i)

```



## 六、实验体会

通过本次系统的设计，我进一步深入了解了 CRM 系统的核心思想及其实现过程，掌握了如何应用关联算法等实现个性化推荐系统，为客户创造更大的价值。

CRM 系统的核心思想，就是以客户为中心提供个性化的服务以降低客户的流失率，通过实现客户效用的最大化获得最大利润。因此，我和队友们抓住这一核心理念，设计了这样一个针对用户肤质的护肤品个性化推荐系统。

在系统的编码过程中，我们基于关联算法实现了推荐功能，能够快速、精确地为用户推荐适合其肤质的护肤品详细信息，通过界面展现给顾客，给用户一种更直观的、个性化的体验感。这一推荐算法的实现，在现实生活中也具有较大的应用价值。

本次系统的设计同时也锻炼了我的数据库应用能力和编程能力，加强了我对 SqlServer 数据库、Python 语言、GUI 界面设计的掌握程度，以及增强了我对关联算法的理解及应用熟练度。

本次实验还提高了我解决问题的能力。小组在实验过程中遇到了一些困难，我和队友们通过积极查阅相关资料、相互沟通交流等方式一一解决，并在此过程中不断地总结提升、共同进步，收获颇丰。

## 七、教师评语

该学生完成了 CRM 系统之推荐算法的实验，（很好/较好/基本）完成了实验任务，算法设计（很好/较好/尚可），实验结果（很好/较好/尚可），实验体会（深刻/较深刻/尚可）。

因此总体评价为\_\_\_\_\_。

教师签字：

2022 年 12 月 1 日