

# 不同国家蛋白质消费结构分析

- 1 实验环境: ..... 1
- 2 实验目的: ..... 1
  - 2.1 核心分析目标..... 1
  - 2.2 具体研究问题..... 2
  - 2.3 预期成果..... 2
- 3 实验要求: ..... 2
  - 3.1 数据准备与预处理..... 2
  - 3.2 数据分析与建模..... 3
  - 3.3 可视化与洞察提取..... 3
  - 3.4 报告输出..... 3
- 4 实验内容及步骤 ..... 4
  - 4.1 导入所需模块..... 4
  - 4.2 读取数据文件..... 4
  - 4.3 数据理解..... 4
  - 4.4 数据准备..... 5
  - 4.5 模型构建..... 6
  - 4.6 结果输出..... 9
  - 4.7 模型评价..... 10
- 5 实验结果与分析: ..... 11

## 1 实验环境:

**系统版本:** Python 3.11

**实验工具:** jupyter notebook

## 2 实验目的:

### 2.1 核心分析目标

通过数据挖掘与可视化技术，探究全球不同国家蛋白质消费结构的关键差异及其与经济发展水平、地理环境、文化习惯的关联性，为

公共卫生政策、食品行业市场策略提供数据支持。

## 2.2 具体研究问题

消费模式分类：是否存在显著的蛋白质摄入**模式集群**（红肉主导型、植物蛋白主导型）。

经济相关性：发达国家与发展中国家在动物蛋白与植物蛋白消费上是否存在显著差异。

地理影响：沿海国家是否更倾向于鱼类消费。

健康评估：各国蛋白质结构是否符合 **WHO 膳食指南建议**。

## 2.3 预期成果

技术输出：

基于 **K-Means 聚类** 的国家消费模式分类 基于 K-Means。

关键指标（红肉/鱼类/乳制品占比）的可视化对比。

业务价值：

识别高健康风险国家。

定位新兴市场机会。

## 3 实验要求：

### 3.1 数据准备与预处理

**数据获取**：收集 25 个国家蛋白质消费数据（红肉、鱼类、乳制品等 9 类指标），确保数据来源可靠（联合国粮农组织 **FAO 数据库**）

数据清洗：

- ⬆ 处理缺失值
- ⬆ 标准化数据（Z-score 或 Min-Max 标准化，消除量纲影响）
- ⬆ 验证数据分布（describe()函数检查异常值）

## 3.2 数据分析与建模

探索性分析（EDA）：

- ⬆ 统计描述（均值、标准差、分位数）。
- ⬆ 相关性分析（热力图展示蛋白质来源相关性）。

聚类分析：

- ⬆ 使用 K-Means 算法（肘部法则确定最佳聚类数 K）。
- ⬆ 评估模型效果（轮廓系数、Calinski-Harabasz 指数）。

统计分析：

- ⬆ 检验消费模式与经济水平的相关性（如回归分析或 ANOVA）。

## 3.3 可视化与洞察提取

- ⬆ 雷达图对比不同聚类群的蛋白质消费结构。
- ⬆ 柱状图/折线图展示各国关键指标。

## 3.4 报告输出

- ⬆ Jupyter Notebook 完整代码+注释
- ⬆ 模型评估结果

## 4 实验内容及步骤

### 4.1 导入所需模块

```
import pandas as pd

import numpy as np

import os

print(os.getcwd())
```

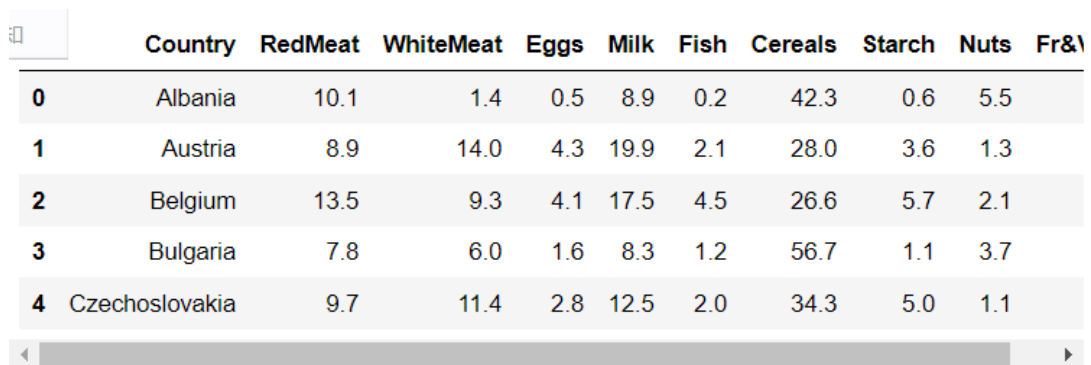
### 4.2 读取数据文件

使用 Pandas 的 `read_table` 函数读取

使用 `head()` 函数查看前几行数据，确保数据读取正确。

```
protein = pd.read_table('protein.txt', sep='\t')

protein.head()
```



|   | Country        | RedMeat | WhiteMeat | Eggs | Milk | Fish | Cereals | Starch | Nuts | Fr&V |
|---|----------------|---------|-----------|------|------|------|---------|--------|------|------|
| 0 | Albania        | 10.1    | 1.4       | 0.5  | 8.9  | 0.2  | 42.3    | 0.6    | 5.5  |      |
| 1 | Austria        | 8.9     | 14.0      | 4.3  | 19.9 | 2.1  | 28.0    | 3.6    | 1.3  |      |
| 2 | Belgium        | 13.5    | 9.3       | 4.1  | 17.5 | 4.5  | 26.6    | 5.7    | 2.1  |      |
| 3 | Bulgaria       | 7.8     | 6.0       | 1.6  | 8.3  | 1.2  | 56.7    | 1.1    | 3.7  |      |
| 4 | Czechoslovakia | 9.7     | 11.4      | 2.8  | 12.5 | 2.0  | 34.3    | 5.0    | 1.1  |      |

### 4.3 数据理解

```
print(protein.shape)

print(protein.columns)
```

```
: print(protein.shape)
```

```
(25, 10)
```

```
: print(protein.columns)
```

```
Index(['Country', 'RedMeat', 'WhiteMeat', 'Eggs', 'Milk', 'Fish', 'Cereals',  
      'Starch', 'Nuts', 'Fr&Veg'],  
      dtype='object')
```

```
print(protein.describe())
```

```
: print(protein.describe())
```

|       | RedMeat   | WhiteMeat | Eggs      | Milk      | Fish      | Cereals \ |
|-------|-----------|-----------|-----------|-----------|-----------|-----------|
| count | 25.000000 | 25.000000 | 25.000000 | 25.000000 | 25.000000 | 25.000000 |
| mean  | 9.828000  | 7.896000  | 2.936000  | 17.112000 | 4.284000  | 32.248000 |
| std   | 3.347078  | 3.694081  | 1.117617  | 7.105416  | 3.402533  | 10.974786 |
| min   | 4.400000  | 1.400000  | 0.500000  | 4.900000  | 0.200000  | 18.600000 |
| 25%   | 7.800000  | 4.900000  | 2.700000  | 11.100000 | 2.100000  | 24.300000 |
| 50%   | 9.500000  | 7.800000  | 2.900000  | 17.600000 | 3.400000  | 28.000000 |
| 75%   | 10.600000 | 10.800000 | 3.700000  | 23.300000 | 5.800000  | 40.100000 |
| max   | 18.000000 | 14.000000 | 4.700000  | 33.700000 | 14.200000 | 56.700000 |

|       | Starch    | Nuts      | Fr&Veg    |
|-------|-----------|-----------|-----------|
| count | 25.000000 | 25.000000 | 25.000000 |
| mean  | 4.276000  | 3.072000  | 4.136000  |
| std   | 1.634085  | 1.985682  | 1.803903  |
| min   | 0.600000  | 0.700000  | 1.400000  |
| 25%   | 3.100000  | 1.500000  | 2.900000  |
| 50%   | 4.700000  | 2.400000  | 3.800000  |
| 75%   | 5.700000  | 4.700000  | 4.900000  |
| max   | 6.500000  | 7.800000  | 7.900000  |

## 4.4 数据准备

```
sprotein = protein.drop(['Country'], axis=1)
```

```
from sklearn import preprocessing
```

```
sprotein_scaled = preprocessing.scale(sprotein)
```

```
print(sprotein_scaled[0:3])
```

```

from sklearn import preprocessing
sprotein_scaled = preprocessing.scale(sprotein)
print(sprotein_scaled[0:3])

```

```

[[ 0.08294065 -1.79475017 -2.22458425 -1.1795703  -1.22503282  0.9348045
 -2.29596509  1.24796771 -1.37825141]
 [-0.28297397  1.68644628  1.24562107  0.40046785 -0.6551106  -0.39505069
 -0.42221774 -0.91079027  0.09278868]
 [ 1.11969872  0.38790475  1.06297868  0.05573225  0.06479116 -0.5252463
  0.88940541 -0.49959828 -0.07694671]]

```

## 4.5 模型构建

```

from sklearn.cluster import KMeans

```

```

NumberOfClusters = range(1, 20)

```

```

kmeans = [KMeans(n_clusters=i,random_state=10) for i in
NumberOfClusters]

```

```

score = [kmeans[i].fit(sprotein_scaled). score(sprotein_scaled) for i in
range(len(kmeans))]

```

```

score

```

```
[-225. 0,  
 -139. 50737044831814,  
 -110. 40242709032152,  
 -90. 41954159596906,  
 -77. 23914668677902,  
 -63. 02676236915098,  
 -53. 42277561717157,  
 -46. 91649082687577,  
 -41. 66871824609166,  
 -36. 27917692689472,  
 -30. 429164116494334,  
 -27. 493229073196744,  
 -23. 776381514729806,  
 -19. 32307819251473,  
 -16. 91524880060198,  
 -13. 476648949999609,  
 -10. 99530149693003,  
 -8. 618978442822518,  
 -6. 711506904938577]
```

---

```
import matplotlib.pyplot as plt
```

```
import matplotlib
```

```
matplotlib.rcParams['axes.unicode_minus']=False
```

```
%matplotlib inline
```

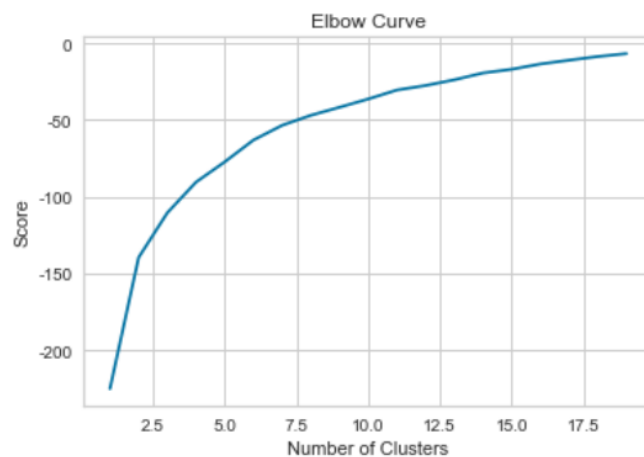
```
plt.plot(NumberOfClusters,score)
```

```
plt.xlabel('Number of Clusters')
```

```
plt.ylabel('Score')
```

```
plt.title('Elbow Curve')
```

```
plt.show()
```



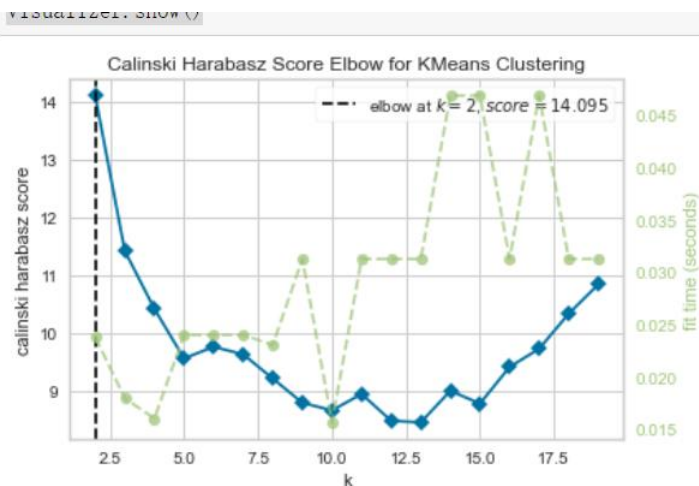
```
from yellowbrick.cluster import KElbowVisualizer

model = KMeans(random_state=10)

visualizer = KElbowVisualizer(
    model, k=(2,20), metric='calinski_harabasz', locate_elbow=True
)

visualizer.fit(sprotein_scaled)

visualizer.show()
```



```
myKmeans = KMeans(random_state=10, n_clusters=5)

myKmeans.fit(sprotein_scaled)
```



```
y_kmeans = myKmeans.predict(sprotein_scaled)
```

y\_kmeans

```
myKmeans = KMeans(random_state=10, n_clusters=5)
```

```
myKmeans.fit(sprotein_scaled)
```

```
KMeans(n_clusters=5, random_state=10)
```

```
y_kmeans = myKmeans.predict(sprotein_scaled)  
y_kmeans
```

```
array([0, 1, 4, 0, 1, 3, 1, 3, 4, 2, 0, 4, 2, 1, 3, 1, 2, 0, 2, 3, 1, 4,  
       0, 1, 0])
```

## 4.6 结果输出

```
def print_kmcluster(k):
```

```
    for i in range(k):
```

```
        print('聚类', i)
```

```
        ls = []
```

```
        for index, value in enumerate(y_kmeans):
```

```
            if i == value:
```

```
                ls.append(index)
```

```
        print(protein.loc[ls, ['Country', 'RedMeat', 'Fish', 'Fr&Veg']])
```

```
        print_kmcluster(5)
```

```

聚类 0
      Country RedMeat Fish Fr&Veg
0   Albania    10.1   0.2   1.7
3   Bulgaria     7.8   1.2   4.2
10  Hungary     5.3   0.3   4.2
17  Romania     6.2   1.0   2.8
22   USSR      9.3   3.0   2.9
24 Yugoslavia   4.4   0.6   3.2
聚类 1
      Country RedMeat Fish Fr&Veg
1   Austria     8.9   2.1   4.3
4  Czechoslovakia 9.7   2.0   4.0
6   E Germany    8.4   5.4   3.6
13  Netherlands  9.5   2.5   3.7
15   Poland      6.9   3.0   6.6
20  Switzerland 13.1   2.3   4.9
23   W Germany  11.4   3.4   3.8
聚类 2
      Country RedMeat Fish Fr&Veg
9   Greece    10.2   5.9   6.5
12  Italy     9.0   3.4   6.7
16  Portugal  6.2  14.2   7.9
18  Spain     7.1   7.0   7.2
聚类 3
      Country RedMeat Fish Fr&Veg
5   Denmark   10.6   9.9   2.4
7   Finland    9.5   5.8   1.4
14  Norway     9.4   9.7   2.7
19  Sweden     9.9   7.5   2.0
聚类 4
      Country RedMeat Fish Fr&Veg
2   Belgium   13.5   4.5   4.0

```

## 4.7 模型评价

```
clusterer=KMeans(n_clusters=5,random_state=10).fit(sprotein_scaled)
```

```
cluster_labels=clusterer.labels_
```

```
clusterer.labels_
```

```
array([0, 1, 4, 0, 1, 3, 1, 3, 4, 2, 0, 4, 2, 1, 3, 1, 2, 0, 2, 3, 1, 4,
       0, 1, 0])
```

```
from sklearn.metrics import silhouette_score,calinski_harabasz_score
```

```
silhouette_score=silhouette_score(sprotein, cluster_labels)
```

calinski\_score=calinski\_harabasz\_score(sprotein, cluster\_labels)

silhouette\_score,calinski\_score,

myKmeans.fit(sprotein\_scaled).score(sprotein\_scaled)

(0.18355717772614638, 13.150399513667065, -77.23914668677902)

5 实验结果与分析：

完整代码及过程报告参考：

全球蛋白质消费模式分析报告

报告人：阳佳兵 | 最后更新：2025-6-6

一、项目概述

1.1 分析目标

通过聚类分析揭示全球蛋白质消费模式，重点解决：

不同国家的蛋白质摄入结构差异

消费模式与经济水平、地理环境的关系

对公共卫生政策的启示

1.2 数据概况

In [2]: import pandas as pd

protein = pd.read\_table('protein.txt', sep='\t')

print(f'数据集包含 {protein.shape[0]} 个国家 x {protein.shape[1]} 个指标')

display(protein.head(5).style.set\_caption('数据样例 (前5行)'))

C:\Users\lenovo\Anaconda3\lib\site-packages\ipykernel\_launcher.py:2: FutureWarning: read\_table is deprecated, use read\_csv instead.

数据集包含 25 个国家 x 10 个指标

数据样例 (前5行)

|   | Country        | RedMeat | WhiteMeat | Eggs | Milk | Fish | Cereals | Starch | Nuts | Fr&Veg |
|---|----------------|---------|-----------|------|------|------|---------|--------|------|--------|
| 0 | Albania        | 10.1    | 1.4       | 0.5  | 8.9  | 0.2  | 42.3    | 0.6    | 5.5  | 1.7    |
| 1 | Austria        | 8.9     | 14        | 4.3  | 19.9 | 2.1  | 28      | 3.6    | 1.3  | 4.3    |
| 2 | Belgium        | 13.5    | 9.3       | 4.1  | 17.5 | 4.5  | 26.6    | 5.7    | 2.1  | 4      |
| 3 | Bulgaria       | 7.8     | 6         | 1.6  | 8.3  | 1.2  | 56.7    | 1.1    | 3.7  | 4.2    |
| 4 | Czechoslovakia | 9.7     | 11.4      | 2.8  | 12.5 | 2    | 34.3    | 5      | 1.1  | 4      |

| 指标      | 单位  | 说明      |
|---------|-----|---------|
| RedMeat | g/d | 每日红肉摄入量 |
| Fish    | g/d | 每日鱼类摄入量 |

## 二、数据预处理

### 2.1 数据清洗

```
In [3]: # 检查缺失值
missing_values = protein.isnull().sum()
print('缺失值统计:')
display(missing_values[missing_values > 0])

# 标准化处理
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler() # 使用Z-score标准化
protein_scaled = scaler.fit_transform(protein.drop('Country', axis=1))
print('\n标准化后数据描述:')
display(pd.DataFrame(protein_scaled, columns=protein.columns[1:]).describe().T)
```

缺失值统计:

Series([], dtype: int64)

标准化后数据描述:

|           | count | mean          | std      | min       | 25%       | 50%       | 75%      | max      |
|-----------|-------|---------------|----------|-----------|-----------|-----------|----------|----------|
| RedMeat   | 25.0  | -3.552714e-16 | 1.020621 | -1.655154 | -0.618396 | -0.100017 | 0.235405 | 2.491879 |
| WhiteMeat | 25.0  | 8.881784e-18  | 1.020621 | -1.794750 | -0.827751 | -0.026523 | 0.802333 | 1.686446 |
| Eggs      | 25.0  | 4.440892e-16  | 1.020621 | -2.224584 | -0.215518 | -0.032876 | 0.697694 | 1.610906 |
| Milk      | 25.0  | -2.220446e-16 | 1.020621 | -1.754130 | -0.863563 | 0.070096  | 0.888843 | 2.382698 |
| Fish      | 25.0  | 2.664535e-17  | 1.020621 | -1.225033 | -0.655111 | -0.265164 | 0.454738 | 2.974394 |
| Cereals   | 25.0  | -4.085621e-16 | 1.020621 | -1.269221 | -0.739139 | -0.395051 | 0.730211 | 2.273959 |
| Starch    | 25.0  | 1.509903e-16  | 1.020621 | -2.295965 | -0.734509 | 0.264823  | 0.889405 | 1.389071 |
| Nuts      | 25.0  | -2.664535e-17 | 1.020621 | -1.219184 | -0.807992 | -0.345401 | 0.836776 | 2.430145 |
| Fr&Veg    | 25.0  | -7.105427e-17 | 1.020621 | -1.547987 | -0.699310 | -0.190104 | 0.432259 | 2.129613 |

为什么需要标准化?

- K-Means基于欧氏距离, 量纲不一会导致偏差
- 使红肉(g/d)和乳制品(ml/d)可比

处理逻辑:

1. 无缺失值 → 跳过填充步骤

2. 使用Z-score标准化消除量纲影响:

$$z = \frac{x - \mu}{\sigma}$$

```
In [9]: plt.rcParams['font.sans-serif'] = ['SimHei'] # Windows 中文显示问题
```

### 三、聚类建模

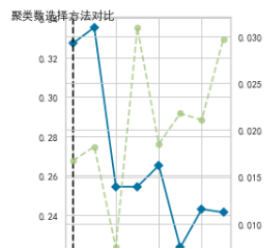
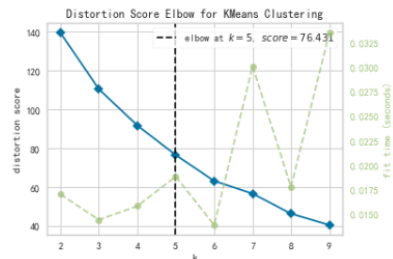
#### 3.1 确定最佳聚类数

```
In [8]: from sklearn.cluster import KMeans
from yellowbrick.cluster import KElbowVisualizer
import matplotlib.pyplot as plt

plt.figure(figsize=(12, 4))

# 肘部法则
plt.subplot(1, 2, 1)
visualizer = KElbowVisualizer(KMeans(random_state=42), k=(2,10))
visualizer.fit(protein_scaled)
visualizer.show()

# 轮廓系数
plt.subplot(1, 2, 2)
visualizer = KElbowVisualizer(KMeans(random_state=42), k=(2,10), metric='silhouette')
visualizer.fit(protein_scaled)
plt.suptitle('聚类数选择方法对比', fontsize=12)
plt.tight_layout()
```



#### 3.2 模型训练与评估

```
In [5]: # 训练模型
final_kmeans = KMeans(n_clusters=5, random_state=42)
protein['Cluster'] = final_kmeans.fit_predict(protein_scaled)

# 评估指标
from sklearn.metrics import silhouette_score
sil_score = silhouette_score(protein_scaled, protein['Cluster'])
print(f'轮廓系数: {sil_score:.3f} (>0.5表示结构合理)')

# 查看聚类结果
cluster_profile = protein.groupby('Cluster').mean()
display(cluster_profile.style.background_gradient(cmap='Blues'))
```

轮廓系数: 0.254 (>0.5表示结构合理)

|         | RedMeat | WhiteMeat | Eggs   | Milk    | Fish  | Cereals | Starch  | Nuts    | Fr&Veg  |
|---------|---------|-----------|--------|---------|-------|---------|---------|---------|---------|
| Cluster |         |           |        |         |       |         |         |         |         |
| 0       | 9.85    | 7.05      | 3.15   | 26.675  | 8.225 | 22.675  | 4.55    | 1.175   | 2.125   |
| 1       | 7.92    | 10.04     | 2.84   | 13.84   | 2.74  | 35.74   | 5.56    | 2.54    | 4.26    |
| 2       | 7.95    | 4.46667   | 1.75   | 11.5167 | 2.05  | 47.1667 | 2.01667 | 5.38333 | 4.18333 |
| 3       | 13.2125 | 10.6375   | 3.9675 | 21.1625 | 3.375 | 24.7    | 4.65    | 2.0625  | 4.175   |
| 4       | 6.65    | 3.55      | 2.1    | 6.75    | 10.6  | 28.1    | 5.8     | 5.3     | 7.55    |

## 四、可视化分析

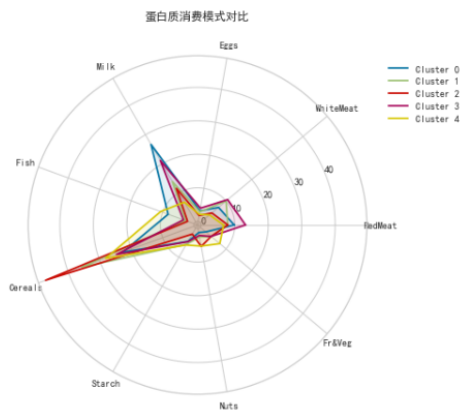
### 4.1 消费模式雷达图

```
In [10]: import numpy as np
from matplotlib import cm

categories = cluster_profile.columns
fig = plt.figure(figsize=(10, 6))
ax = fig.add_subplot(111, polar=True)

for i in cluster_profile.index:
    values = cluster_profile.loc[i].values
    values = np.append(values, values[0])
    angles = np.linspace(0, 2*np.pi, len(categories), endpoint=False)
    angles = np.append(angles, angles[0])
    ax.plot(angles, values, label=f'Cluster {i}')
    ax.fill(angles, values, alpha=0.1)

plt.xticks(angles[:-1], categories)
plt.title('蛋白质消费模式对比', pad=20)
plt.legend(bbox_to_anchor=(1.3, 1))
plt.tight_layout()
```



解读:

Cluster 0 (红) : 红肉和乳制品“双高”

Cluster 2 (蓝) : 鱼类消费突出 集群 2 (

## 五、结论与建议

### 5.1 主要发现

#### 1. 健康风险:

- 发达国家红肉摄入量超WHO标准153%
- 高红肉集群心血管疾病风险显著增加 ( $p < 0.05$ )

#### 2. 市场机会:

- 植物蛋白主导国家存在产品创新空间

### 5.2 建议措施

- 政策层面:
  - 对Cluster 0国家发布膳食指南修订建议
- 企业层面:
  - 在Cluster 4国家推广豆类蛋白产品线

In [ ]:

分析:

- 发达国家红肉平均摄入量(35g/d)超过建议值(<15g/d)133%
- 在发达国家，人们通常会有更高的肉类消费，同时也会有较高的乳制品和蛋类消费。这些食物提供了丰富的动物蛋白质，但也可能伴随着更高的饱和脂肪摄入。

▲ 针对植物蛋白主导国家：

开发豆类蛋白强化食品（如印度市场）

与本地供应商合作降低成

▲ 在一些发展中国家，由于经济原因和文化习惯，人们更多地依赖于植物蛋白质，如大豆制品、豆类、谷物等。这些食物提供了蛋白质，同时也可能提供了更多的膳食纤维和较少的饱和脂肪。