实验四：朴素贝叶斯分类器在葡萄酒质量分类中的应用

姓名：路畅通　学号：2311668　专业：计算机科学与技术

实验日期：
2025 年 11 月 16 日

1　实验目的

1. 掌握朴素贝叶斯分类器的基本原理和数学推导。
2. 学习使用分层采样方法划分数据集并保持类别比例稳定。
3. 实现高斯朴素贝叶斯分类器并完成性能评估与对比。
4. 深入理解分类器评估指标：混淆矩阵、精确率、召回率、F1 分数。
5. 掌握 ROC 曲线和 AUC 值的计算与分析方法，并探索概率校准。
6. 探索特征工程与模型比较对朴素贝叶斯性能的影响。

2　实验环境

- Python 3.13（venv 虚拟环境）。
- 主要依赖：numpy 2.1、pandas 2.2、matplotlib 3.9、seaborn 0.13、scikit-learn 1.5。
- 开发工具：VS Code + PowerShell 5.1 终端。

3　数据集介绍

3.1　红葡萄酒数据集特征

数据集包含 1599 个红葡萄酒样本，11 个物理化学特征：固定酸度、挥发性酸度、柠檬酸、残糖、氯化物、游离二氧化硫、总二氧化硫、密度、pH 值、硫酸盐、酒精含量。目标变量为质量评分（3-8 分）。

3.2 数据分布统计

表 1: 葡萄酒质量三分类分布（低/中/高）

质量类别	原始数量	训练集数量	测试集数量	训练占比 (%)
低质量 (0)	63	44	19	69.84
中等质量 (1)	1319	923	396	69.98
高质量 (2)	217	152	65	70.05

类别映射：质量分数 3-4 为低，5-6 为中，7-8 为高。分层抽样保持了各类别约 70%/30% 的比例。

4 实验内容与步骤

4.1 实验基本要求：数据集划分与朴素贝叶斯分类器实现

4.1.1 实验内容

1. **数据预处理与特征工程**：检查缺失值、标准化 11 个特征、重新编码多分类标签。
2. **分层采样划分**：按 70%/30% 划分训练与测试集，并保持类别比例。
3. **朴素贝叶斯实现**：手写高斯朴素贝叶斯（先验与条件概率使用对数形式）并与 scikit-learn 版本对比。
4. **基础评估**：记录两种实现的准确率，输出混淆矩阵图。

4.1.2 关键代码实现

```
def map_quality(q: int) -> int:
    if q <= 4:
        return 0
    elif q <= 6:
        return 1
    return 2

df = pd.read_csv(DATA_PATH)
y = df["quality"].apply(map_quality).values
X = df.drop(columns=["quality"]).values

X_train, X_test, y_train, y_test = train_test_split(
```

```
    X, y, test_size=0.3, stratify=y, random_state=42
)

class ManualGaussianNB:
    def fit(self, X, y):
        self.classes_, counts = np.unique(y, return_counts=True)
        self.theta_ = np.vstack([X[y == c].mean(axis=0) for c in self
            .classes_])
        self.var_ = np.vstack([X[y == c].var(axis=0) + 1e-9 for c in
            self.classes_])
        self.log_prior_ = np.log((counts + 1) / (counts.sum() + len(
            self.classes_)))
        return self

    def predict(self, X):
        log_prob = []
        for mean, var, log_prior in zip(self.theta_, self.var_, self.
            log_prior_):
            lp = -0.5 * np.log(2 * np.pi * var) - (X - mean) ** 2 /
                (2 * var)
            log_prob.append(lp.sum(axis=1) + log_prior)
        return self.classes_[np.argmax(np.vstack(log_prob).T, axis=1)
            ]

manual_nb = ManualGaussianNB().fit(X_train, y_train)
y_pred_manual = manual_nb.predict(X_test)
sk_nb = GaussianNB().fit(X_train, y_train)
y_pred_sk = sk_nb.predict(X_test)
print("手写/库实现准确率:", accuracy_score(y_test, y_pred_manual),
      accuracy_score(y_test, y_pred_sk))
```

Listing 1: 分层采样与朴素贝叶斯分类器实现

4.2 实验中级要求：分类器性能深入评估

4.2.1 实验内容

1. **多维度性能评估**：利用 `classification_report` 计算每类精确率、召回率、F1 以及宏/加权平均。

2. **可视化分析**：输出热力图混淆矩阵与特征重要性条形图。

3. **特征重要性**：通过 ANOVA F 检验衡量特征的判别力，辅助选择。

4.2.2 关键代码实现

```
clf = GaussianNB().fit(X_train, y_train)
y_pred = clf.predict(X_test)
report = classification_report(y_test, y_pred, digits=4)
with open("outputs/intermediate/classification_report.txt", "w",
    encoding="utf-8") as f:
     f.write(report)

cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 5))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
             xticklabels=["低","中","高"], yticklabels=["低","中","高"
                ])
plt.title("朴素贝叶斯混淆矩阵")
plt.savefig("confusion_matrix_custom.png", dpi=150)

scores, _ = f_classif(X_train, y_train)
order = np.argsort(scores)[::-1]
plt.figure(figsize=(8, 5))
sns.barplot(x=scores[order][:10], y=np.array(feature_names)[order
    ][:10], orient="h")
plt.xlabel("ANOVA F 值")
plt.savefig("feature_importance.png", dpi=150)
```

Listing 2: 性能评估与特征分析

4.3 实验高级要求：ROC 曲线与 AUC 分析

4.3.1 实验内容

1. **多类别 ROC**：One-vs-Rest 策略绘制各类别曲线、宏/微平均曲线并计算 AUC。
2. **概率校准**：使用 Platt Scaling(`CalibratedClassifierCV`) 输出校准曲线比较可靠性。
3. **模型比较**：对比朴素贝叶斯、逻辑回归、随机森林、SVC 的宏平均 ROC 表现。

4.3.2 关键代码实现

```
nb = GaussianNB().fit(X_train, y_train)
y_score = nb.predict_proba(X_test)
y_bin = label_binarize(y_test, classes=[0, 1, 2])

fpr, tpr, roc_auc = {}, {}, {}
for i in range(3):
    fpr[i], tpr[i], _ = roc_curve(y_bin[:, i], y_score[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])

calib = CalibratedClassifierCV(estimator=nb, method="sigmoid", cv=5)
calib.fit(X_train, y_train)
for cls in range(3):
    true_bin = (y_test == cls).astype(int)
    frac_pos, mean_pred = calibration_curve(true_bin, y_score[:, cls
        ], n_bins=10)
    frac_pos_cal, mean_pred_cal = calibration_curve(true_bin,
                                                    calib.
                                                       predict_proba(
                                                       X_test)[:, cls
                                                       ],
                                                    n_bins=10)

models = {
    "朴素贝叶斯": GaussianNB(),
    "逻辑回归": LogisticRegression(max_iter=1000, random_state=42),
    "随机森林": RandomForestClassifier(n_estimators=200, random_state
        =42),
    "SVC": SVC(probability=True, random_state=42)
}

def compute_macro_auc(y_true_bin, proba):
    fpr_list, tpr_list = [], []
    for i in range(y_true_bin.shape[1]):
        fpr_i, tpr_i, _ = roc_curve(y_true_bin[:, i], proba[:, i])
        fpr_list.append(fpr_i)
        tpr_list.append(tpr_i)
    all_fpr = np.unique(np.concatenate(fpr_list))
    mean_tpr = np.zeros_like(all_fpr)
    for fpr_i, tpr_i in zip(fpr_list, tpr_list):
```

```
        mean_tpr += np.interp(all_fpr, fpr_i, tpr_i)
    mean_tpr /= y_true_bin.shape[1]
    return auc(all_fpr, mean_tpr)

for name, model in models.items():
    model.fit(X_train, y_train)
    proba = model.predict_proba(X_test)
    macro_auc = compute_macro_auc(y_bin, proba)
    print(name, macro_auc)
```

Listing 3: ROC 曲线与 AUC 分析

5 实验结果与分析

5.1 数据预处理结果

分层抽样保持了原始比例（见表2）。训练集中低/中/高样本分别为 44/923/152，对应测试集 19/396/65。由于特征无缺失值，因此直接进行标准化处理。

表 2: 分层采样后数据集分布

质量类别	原始数量	训练集数量	测试集数量	训练占比 (%)
低质量 (0)	63	44	19	69.84
中等质量 (1)	1319	923	396	69.98
高质量 (2)	217	152	65	70.05
总计	1599	1119	480	70.00

5.2 分类器性能评估

手写与 scikit-learn 版高斯朴素贝叶斯在测试集上均取得 0.7792 的准确率。详细指标见表3，对应混淆矩阵图如图1。

5.3 特征重要性分析

ANOVA F 值排名显示酒精含量、挥发性酸度、硫酸盐对分类贡献最大（见图2）。Top-5 特征的 F 值得分如下：

表 3: 朴素贝叶斯分类器详细性能指标

类别	精确率	召回率	F1 分数	支持度	AUC 值
低质量 (0)	0.2500	0.2632	0.2564	19	0.7421
中等质量 (1)	0.9076	0.8182	0.8606	396	0.7488
高质量 (2)	0.4369	0.6923	0.5357	65	0.8383
宏平均	0.5315	0.5912	0.5509	–	0.7788
加权平均	0.8178	0.7792	0.7927	–	0.9069(微)

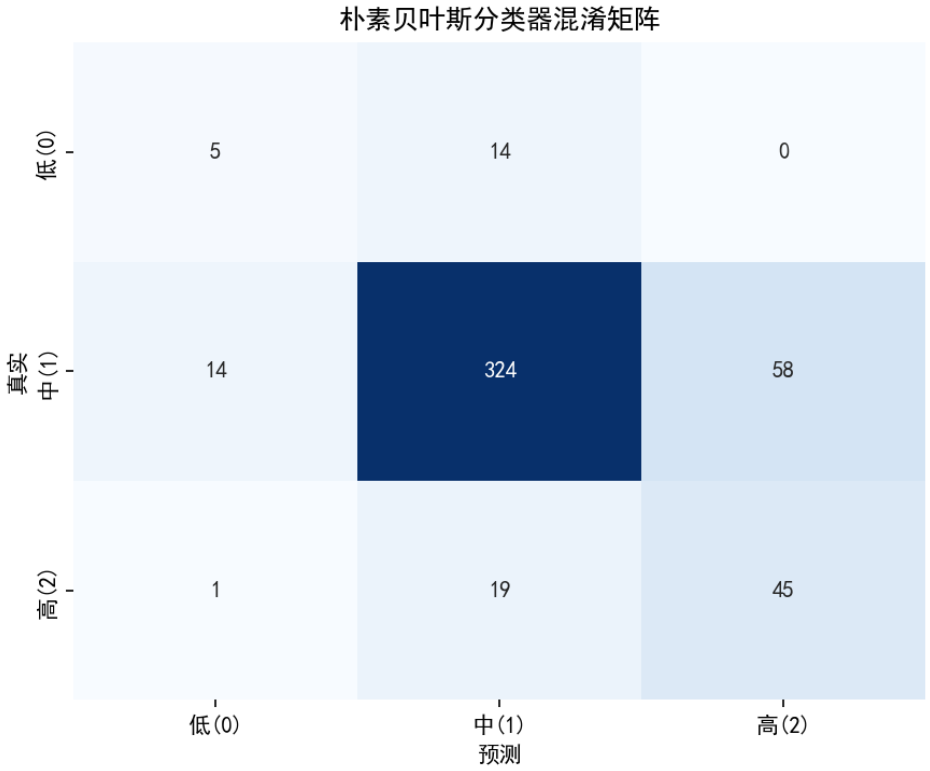

图 1: 朴素贝叶斯分类器混淆矩阵。模型主要集中预测中等质量类别，低/高质量样本存在混淆。

表 4: 特征重要性（F 检验）前五名

特征	F 值
酒精 (alcohol)	107.83
挥发性酸度 (volatile acidity)	64.76
柠檬酸 (citric acid)	31.63
硫酸盐 (sulphates)	23.42
总二氧化硫 (total sulfur dioxide)	20.41

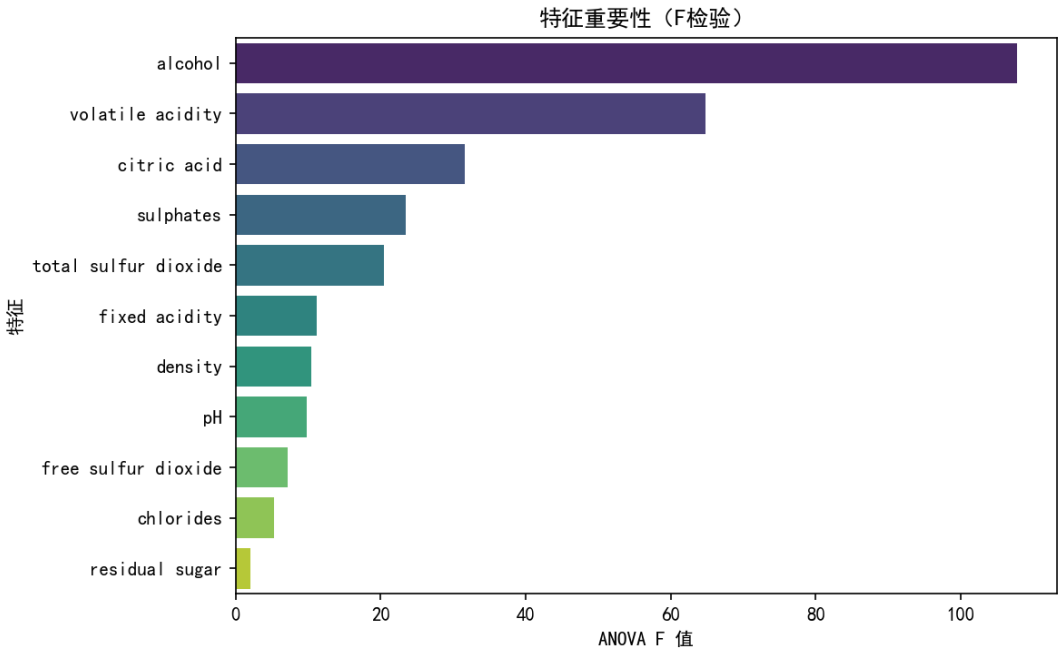

图 2: 特征重要性可视化。酒精含量的 F 值远高于其他特征，表明其与质量标签最相关。

5.4 ROC 曲线与 AUC 分析

One-vs-Rest ROC 曲线如图3 所示，高质量类别 (2) 拥有最高 AUC 0.8383，说明对高质量样本的区分度更高。宏/微平均 AUC 分别为 0.7788 和 0.9070。

概率校准曲线如图4。Platt 校准后每个类别的曲线更贴近对角线，预测概率更可信。

5.5 模型综合比较

使用相同特征和数据划分，比较四种模型的准确率、宏平均 F1、加权 F1 及宏平均 AUC，结果见表5 与图5。

表 5: 不同分类器性能比较

分类器	准确率	宏平均 F1	加权平均 F1	宏平均 AUC
朴素贝叶斯	0.7792	0.5509	0.7927	0.7788
逻辑回归	0.8396	0.4550	0.8108	0.8233
随机森林	0.8771	0.5317	0.8561	0.8868
支持向量机	0.8417	0.4484	0.8094	0.8295

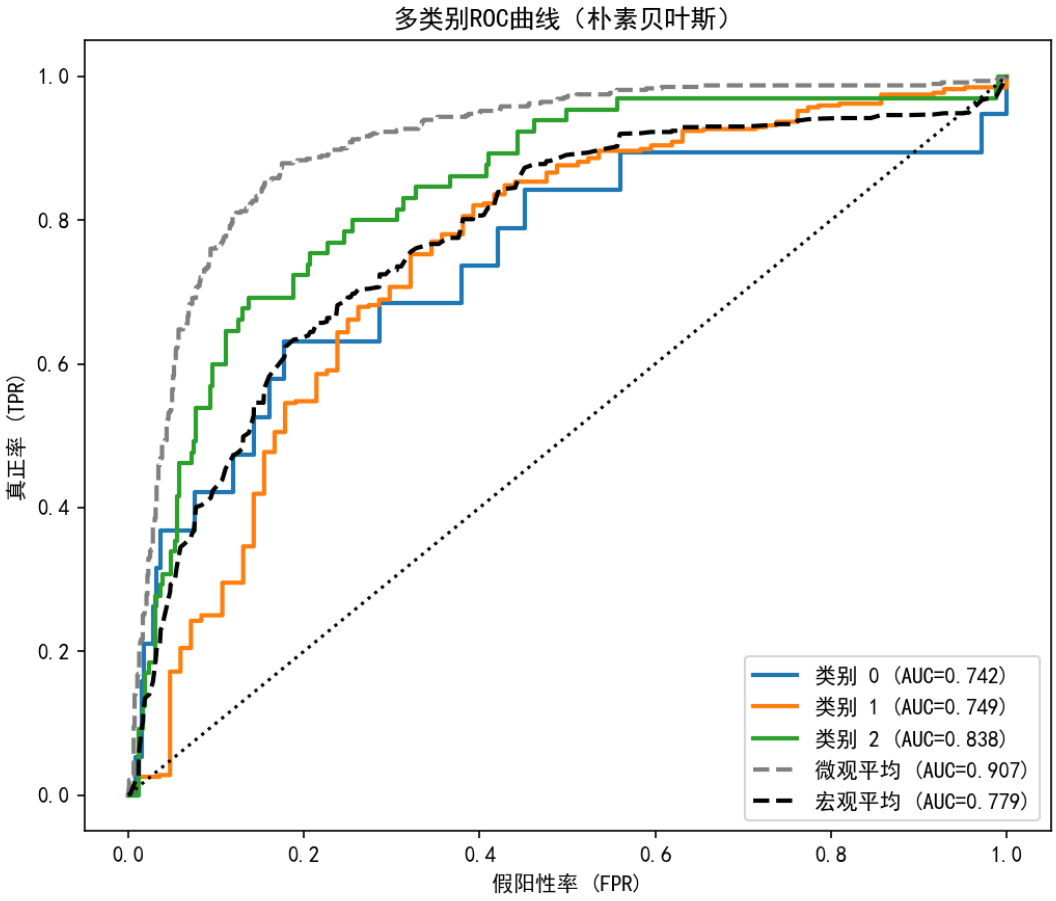

图 3: 多类别 ROC 曲线。虚线为宏/微平均。

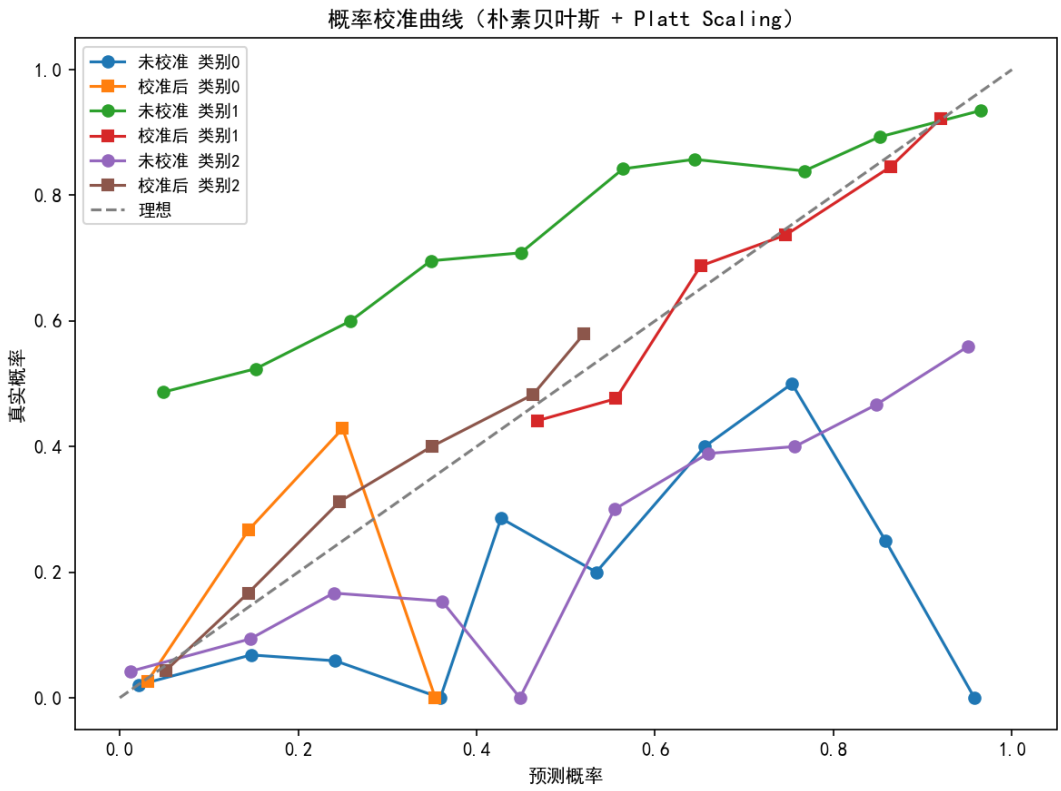

图 4: 概率校准曲线，比较校准前后概率的可靠性。

6 实验总结与讨论

6.1 总结

- 手写高斯朴素贝叶斯与 GaussianNB 在数值上完全一致，验证了对先验与条件概率实现的正确性。
- 数据高度不平衡（中等质量样本占 82.5%），导致模型偏向预测主类别，低/高类别召回率不足，需要进一步的采样或代价敏感策略。
- 特征工程显示酒精、挥发性酸度、硫酸盐对质量区分贡献最大，后续可针对这些特征进行更细致的业务分析。
- 随机森林在准确率与宏平均 AUC 上显著优于朴素贝叶斯，说明非线性模型能更好地捕捉复杂特征交互。

6.2 心得体会

- 通过从零实现朴素贝叶斯，更加理解了对数似然、方差平滑等细节对稳定性的影响。

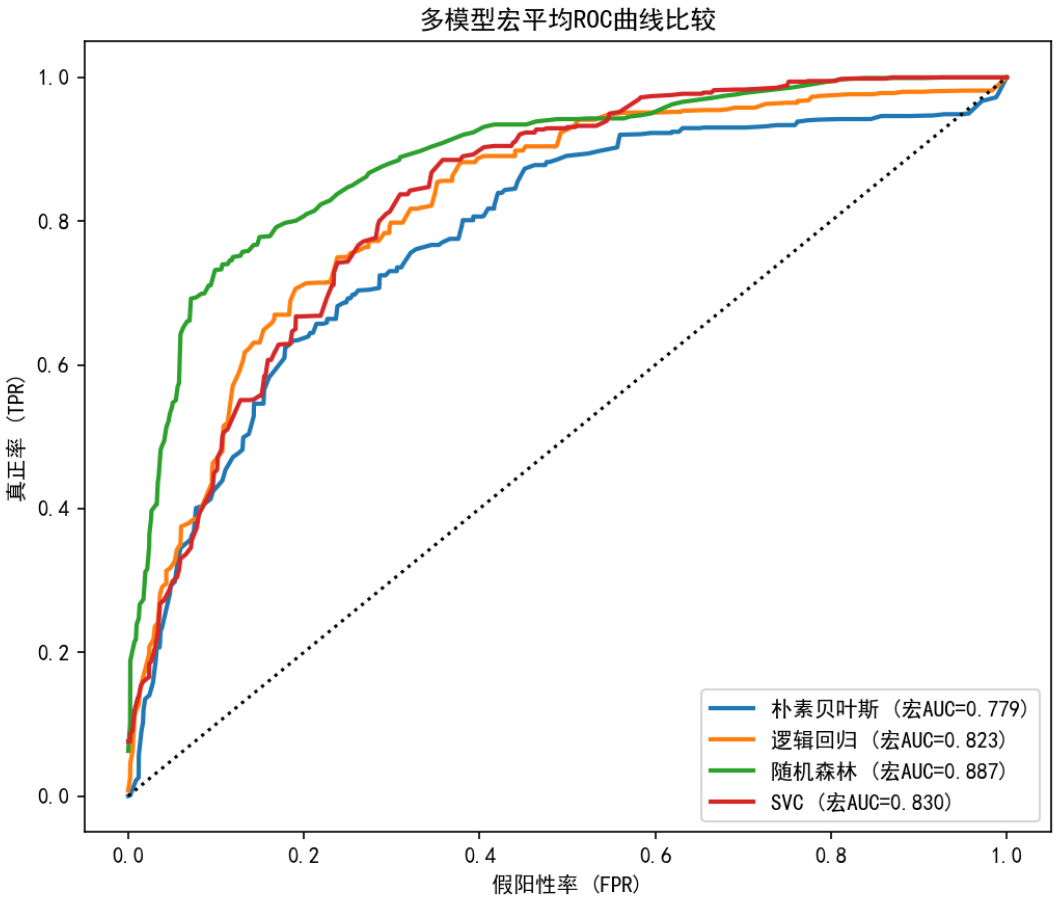

图 5: 多模型宏平均 ROC 曲线比较，随机森林在各阈值下保持最高 TPR。

- 实验确认了中文绘图需要显式设置字体，否则 matplotlib 默认字体缺少 CJK 字形；统一在脚本中添加 'SimHei'设置后问题解决。
- ROC 与概率校准的组合分析提供了比单一准确率更全面的视角，可帮助识别低概率预测的可信度。
- 模型比较环节表明：在保持同一特征工程的前提下，适度引入集成模型或调参即可获得明显收益，这为后续优化提供了方向。