“弗洛伊德之死”的统计学视角再发现

——基于美国警方致命性枪击事件数据的种族偏见分析

摘要：

近期美国警察暴力执法事件所导致的“弗洛伊德之死”及相关的种族歧视问题受到了国际社会的广泛讨论与关注。本研究从华盛顿邮报发布的近年美国各州警察执法致命性枪击事件数据出发，运用主成分分析、判别分析及典型相关分析等统计学方法，分析暴力执法中种族偏见现象的相关因素，发现城市整体性特征作用明显超过案件个体性特征，其中城市文化传统起到最为重要的作用，侧面印证了警察执法中的种族偏见的历史渊源与棘手性。

目录：

1 研究问题与背景描述

种族歧视问题一直是全世界讨论的焦点话题之一，而近期美国警察暴力执法事件所导致的“弗洛伊德之死”更是引发了国际社会的广泛讨论与关注。对于该类事件，人们往往习惯于将之主观判断为美国警察执法过程中种族歧视的代表，但缺乏有力的数据证据与深入的分析，暴力执法现象中的种族偏见与哪些因素有关，包含种族在内的诸多因素如何相互作用，个体事件与城市文化又有怎样的关联，这些问题对于我们理解并改善警察暴力执法中的种族歧视问题十分关键，值得我们利用数据与统计学方法进行分析与进一步思考。

2 数据介绍与预处理

2.1 数据集描述

本研究所使用数据集下载自 Kaggle 平台“Fatal Police Shootings in the US”，为华盛顿邮报整理发布的 2015 年-2017 年间美国各州发生的所有警方致命性射击案件数据汇总，总计 2535 起，共有事件 ID、日期、死因、犯者武器、种族、所在城市等 14 个指标被记录，各指标的具体含义与取值将在 2.3 部分经预处理筛选后详细展开。此外，对应于事件发生的城市，该数据集还包含了由美国统计局发布的各城市贫困率、高中毕业率、家庭收入中位数及各种族人口占比四类城市统计指标。将该部分与事件指标相整合，得到初步的二十余个变量。

2.2 探索式数据分析

为了更好地了解数据特点，辅助进一步分析，我们对射击案件数据进行探索式数据分析（EDA），得到的结果如**图 1** 所示。综合图 1 的结果，可以得到以下几条结论：

1） 死者所属种族以白人、黑人与西班牙裔为主，占总人数的 95.8%以上，印第安人、亚裔与其他种族所占比例很小。而对于死亡人数占种族总人口的比例，黑人明显高于其他种族，其次依次是印第安人、西班牙裔、白人与亚裔。由此可以初步得出种族偏见的存在，但仍需进一步分析其他因素的影响。

2） 在年龄分布上，呈现右偏分布，死者年龄集中在 20-45 岁，且呈现不同种族间的异质性：白人死者的年龄普遍高于黑人与西班牙裔，其平均年龄 40 岁明显高于黑人的 32 岁及西班牙裔的 33 岁，说明年龄是影响死者种族分布的一个重要因素。

3） 在地域分布上，加州、得克萨斯州和佛罗里达州是死亡人数最多的三个州，此外，各州死亡总人数分布与黑人死亡人数分布略有差异，在南部与东北部死亡人数黑人占比明显更高；而在中部与西北部则较低，这说明种族与地域是相互影响的，此外，洛杉矶、菲尼克斯、休士顿、芝加哥等城市死亡人数最多。

4） 在性别分布上，男性占 95%以上，远高于女性；在死者所携武器上，枪支占到近 60%，但仍有相当比例（约 10%）死者未携武器，这说明警察暴力执法现象的存在。

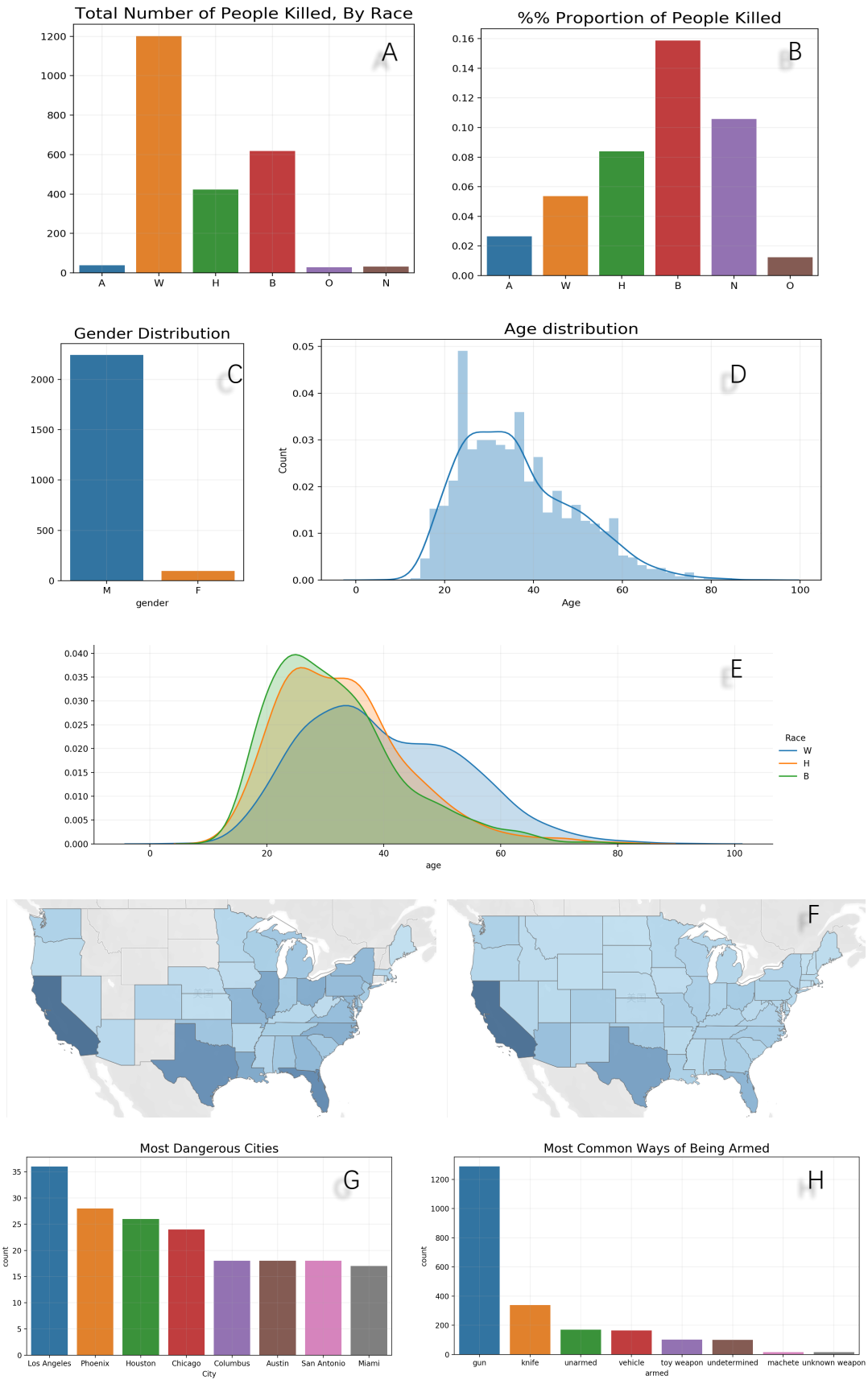

图 1：探索式数据分析结果（图 A：各种族死亡人数分布（A：亚裔；W：白人；B：黑人；H：西班牙裔；N：印第安人；O：其他，下同）；图 B：各种族死亡人数占种族总人口比例（以万分数

记)；图 C：死亡人数性别分布（M：男性；F：女性)；图 D：死亡人数年龄分布；图 E：不同种族死亡人数年龄分布；图 F：各州死亡人数（左：黑人死亡人数；右：总死亡人数)；图 G：死亡人数最多的前八个城市；图 H：死者所携武器，选取出现次数最多的前八个展示）

2.3 数据变换与预处理

将城市数据与案件数据整合，删去编号、日期等无关变量，同时删去州、城市两个变量，避免与城市属性变量重复，得到最终的 16 个变量。之后，对于每一条观测的缺失值进行处理，删去缺失种族信息的观测，将缺失的年龄补为全部观测的年龄均值，最后删去余下的少量含缺失值观测，得到最终 2053 条观测数据，对应的 16 个变量名称及含义如**表 1** 所示：

表 1：16 个变量的名称及含义

名称	含义	名称	含义
armed	所携武器	poverty_rate	城市贫困率
age	年龄	share_white	白人占比
gender	性别	share_black	黑人占比
signs_of_mental_illness	有无精神疾病	share_native_american	印第安占比
threat_level	威胁程度	share_asian	亚裔占比
flee	逃跑方式	share_hispanic	西班牙裔占比
body_camera	有无监控	Median.Income	收入中位数
race	种族	percentage_completed_hs	高中毕业率

其中,armed, gender, signs_of_mental_illlness, threat_level, body_camera,flee, race 为类别型变量，其余均为数值型变量，将其中的二值变量转化为 0/1，将有序的类别型变量 threat_level,flee 转化为连续自然数，余下 race 和 armed 两个无序的类别变量。由于 race 在后续分析中多作为类别标签，armed 包含 59 种不同的类别，若处理为独热编码将大大增加变量维度，产生干扰，因而在这里我们直接使用 LaborEncoder 函数转化为 0-58 的自然数，毫无疑问，这种直接转换将对结果产生一定程度的干扰，也是未来的主要改进方向之一。此外，由于各变量之间尺度差异较大，虽然在一些分析中不会对结果产生影响，但为了统一以及便于比较各变量的贡献,在后续研究中,我们统一使用标准化后的数据进行分析。

3 分析方法设计与实验结果

3.1 主成分分析与降维

为了更好地理解各变量之间的关系，提取主要成分进行降维可视化，我们首先采用主成分分析（PCA)。由于变量 share_white, share_black, share_hispanic, share_asian 与 share_native_american 存在较为明显的共线性关系，结合 EDA 得出的死亡人数所属种族分布，我们仅保留 share_black 与 share_hispanic 加入主成分分析。所得“崖底碎石图”如

图 2 所示：

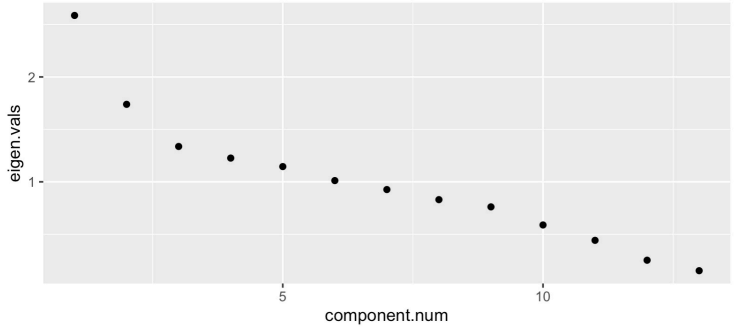

图 2：主成分分析崖底碎石图

由崖底碎石图可以看出，在第三主成分处出现肘点，但与此同时，在该实际数据中，PCA 的效果不甚理想。除前两主成分方差解释比例相对较高外，其余主成分均有一定的解释比例，但未能呈现很集中的分布和明显的截断。各主成分方差解释比例及各变量前三主成分系数如**表 2** 所示：

表 2：主成分分析结果

变量	PC1	PC2	PC3		解释方差比例（%）
armed	0.0140	**-0.4329**	0.2492	PC1	**19.90**
age	0.1400	**0.3641**	-0.2252	PC2	**13.38**
gender	-0.0711	-0.0401	0.1422	PC3	**10.29**
race	0.2681	**0.3322**	-0.1101	PC4	9.43
signs_of_MI	0.1481	**0.1935**	-0.1700	PC5	8.81
threat_level	-0.0067	**-0.3647**	0.0451	PC6	7.78
flee	0.0476	**0.3204**	-0.2386	PC7	7.13
body_camera	-0.0117	0.0062	-0.0549	PC8	6.39
poverty_rate	**-0.5392**	0.1889	0.0223	PC9	5.85
share_black	**-0.3226**	0.1038	0.4145	PC10	4.53
share_hispanic	-0.1929	-0.3722	-0.6390	PC11	3.40
Median.Income	**0.4607**	-0.3255	-0.1480	PC12	1.94
percentage_hs	**0.4843**	0.0625	0.4017	PC13	1.17

根据**表 2** 中的前两主成分系数，结合系数数值相对较大的变量含义（已加粗）我们可以得到如下的直观含义解释：第一主成分代表案件发生城市的发达程度（经济发展与教育水平）；第二主成分代表死者个人特征（种族、精神状况、年龄）及个体案件激烈程度（所携武器、逃跑方式），它们均具有比较明确的实际含义。

为了研究这些变量之间相互作用及与我们最为关注的变量——种族之间的关系，根据各条观测的主成分得分，我们以 race 为类别标签对所有两千多条数据进行降维可视化，不同种族的数据点以不同颜色与形状表示。所得的二维与三维图像如**图 3** 所示：

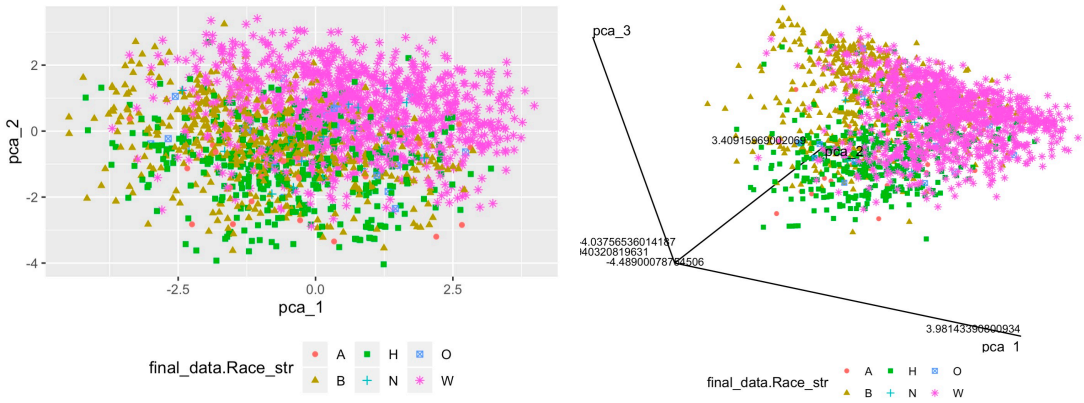

图 3：依据主成分进行降维可视化（左：降至二维；右：降至三维）

可以看出，在三维下的类别区分度略好于二维，占比最大的三类（B, W, H）已可以看出一定程度的分离，但均呈现较严重的混淆。这说明对于 race 作为分类变量，PCA 的效果不是很理想，由于总变量数不超过 20，我们在后续判别分析中直接使用原始变量。

3.2 判别分析

在本部分，我们将重点关心的变量——race 作为类别标签进行判别分析，从判别准确率、混淆矩阵情况及各变量对判别结果重要程度等方面进行探讨。对于所有观测，我们按照 7:3 的比例将之划分为训练集与测试集，分别计算训练集与测试集的 APER 作为模型评价指标，其中，验证集 APER 可以有效避免过拟合带来准确率偏高的影响，准确评估模型表现；测试集与验证集之差则反映了过拟合现象是否严重。

3.2.1 LDA 与 QDA

LDA 与 QDA 是最经典的判别分析方式，LDA 是一类线性判别器，QDA 则放松了 LDA 中各类等方差的假设，并由求解过程得到非线性结构。对于本数据，LDA 与 QDA 在训练集与测试集分别的 APER 值如**表 3** 所示：

表 3：LDA 与 QDA 判别结果

APER	训练集	测试集
LDA	**0.3278**	**0.3588**
QDA	0.3319	0.4237

可以看出，LDA 无论在训练集还是测试集，APER 都明显低于 QDA，代表判别准确率更高，然而，在我们一般理解中，假设更少的非线性 QDA 应该取得更好的结果，具体原因还有待进一步探讨。从混淆矩阵来看，QDA 的高错误率主要源于将许多 W 类及 B 类观测归为 H 类。

为了更加直观地呈现判别结果，在**图 4** 中展示出 LDA 判别的混淆矩阵，其中第 i 行第 j

列的数值代表真实标签为 i 类的观测被分为 j 类的比例。由图可以看出，真实标签为 B 和 H 的观测被错判为 W 的可能性较高，而 W 类的分类正确率相对较高。此外，A，N，O 等观测数本身很少的类的判别波动性比较大，给分析带来一定困难。

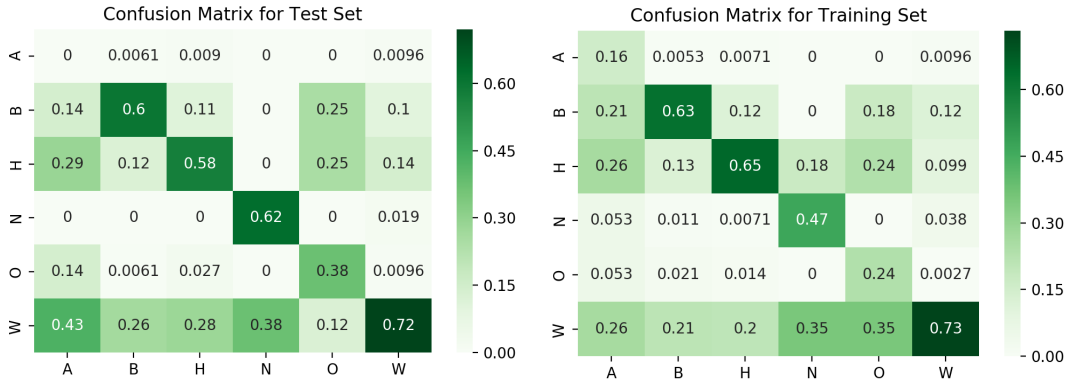

图 4：LDA 判别的混淆矩阵（左：测试集；右：训练集）

将 LDA 判别结果与真实类别标签结合，以 **3.1** 所得出的前两主成分进行降维可视化，所得结果呈现于**图 5** 中，其中不同颜色代表 LDA 判别标签类，不同形状代表真实类别标签，可以看出，多数观测点呈现正确的分类，混淆主要发生在图中部类别各交错的位置。此外，相比 PCA 之下的混淆情况，判别结果的混淆程度要轻许多。

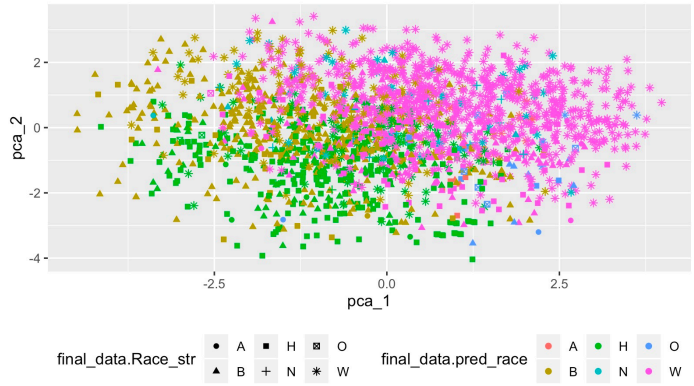

图 5：依据主成分对观测点 LDA 判别结果及真实标签降维可视化

3.2.2 其他判别方法

根据以上分析，LDA 与 QDA 都没有呈现较好的分类效果，这可能与它们的判别分类面较为简单，不适合于本任务数据有关，因此，我们尝试更为灵活的随机森林（RF）和支持向量机（SVM）模型进行判别分类。

RF 以决策树为基本单元，采用集成的思想，通过引入随机因素所生成的多棵树结果的投票作为最终分类结果，解决了单棵决策树易受噪声扰动的缺点，具有很强的拟合能力与灵活性。树的个数的选取对于 RF 模型十分重要，一般采用交叉验证的方式，选取验证集表现最佳之处作为模型中树的棵数。以 25 为间隔分别尝试 25-250 之间的模型拟合效果，如**图 6**

左所示，在棵数为 175 时，测试集 APER 达到最小值 0.3474，因而选取棵数 175 作为最终模型。同时在**图 6** 右中绘出各变量对判别的重要性排序，可以看出所属城市种族比例对于 race 的预测最为重要，年龄紧随其后，这与 EDA 部分发现的白人死者年龄普遍高于黑人与西班牙裔是一致的。此外，案发城市的各特征变量对于判别种族的重要性普遍高于个体案件及死者个人特征，这在一定程度上说明警察暴力执法的种族偏见是一地域性的、群体性的现象，而非偶然的、个体性的现象。

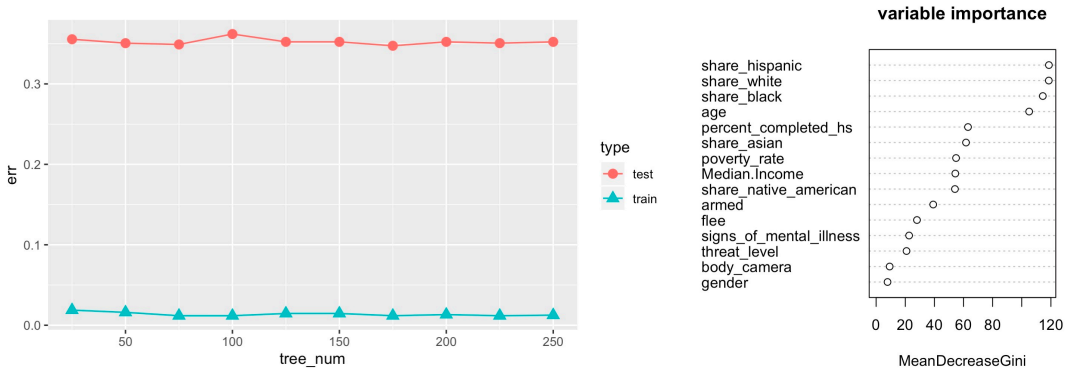

图 6：随机森林模型结果（左：不同棵数下的 APER；右：各变量重要性）

SVM 则引入最大间隔的思想，使用 hinge 损失函数进行优化。kernel 的引入与构造则突破了线性框架，大大提升了 SVM 模型的灵活性与拟合能力。这里采用广为使用的高斯核函数带入模型，得到测试集与训练集 APER 分别为 0.3523 及 0.2526。除 APER 外，这里引入一个新的评价指标：AUC，即准确率-召回率组成的 ROC 曲线下方面积，RF 和 SVM 模型的 ROC 曲线及 AUC 值如**图 7** 所示，在新的评价指标下，SVM 的表现略好于 RF，但是对于一个六分类任务，不到 0.7 的 AUC 值仍不甚理想，这可能与特征不足有关。

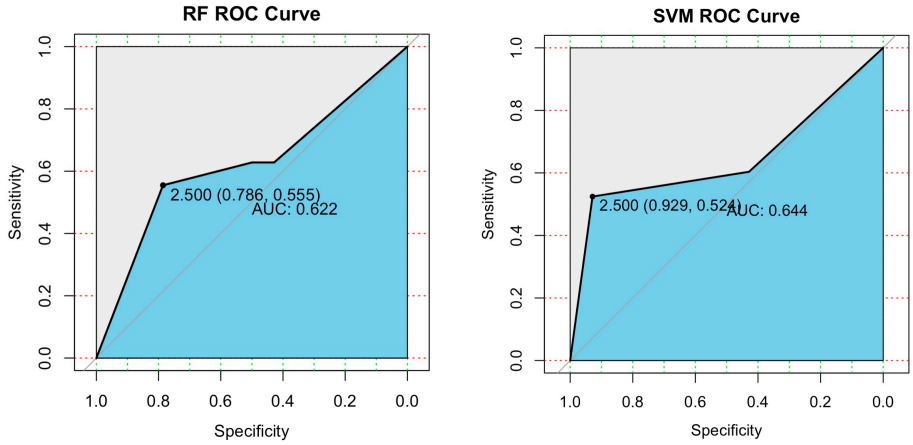

图 7：模型的 ROC 曲线与 AUC 值（左：RF；右：SVM）

表 4 给出 RF，SVM 的判别结果及与 LDA 的对比。可以发现，RF 与 SVM 在测试集的表现均优于 LDA，其中 RF 的拟合能力很强，在训练集错误率接近于 0，产生了较严重的过拟合现象。此外，它们的效果均仍有一定提升空间。

表 4：LDA、RF 与 SVM 的判别结果

APER	LDA	RF	SVM
训练集	0.3278	**0.0118**	0.2526
测试集	0.3588	**0.3474**	0.3523

3.2.3 讨论与小结

在本部分，我们主要做以下两点讨论与改进。首先，针对各类类别不平衡对分类带来的困扰，我们删去所有 N, O, A 类观测，对余下的 1970 条观测分别运用 LDA，RF 及 SVM 模型进行判别分析，结果如**表 5** 所示。由于少了三类罕见类的干扰，三模型的效果相比于六分类问题均有明显提升，以 LDA 效果为最佳，SVM 次之。值得一提的是，即使考虑余下三个被舍弃的类别全部错分，APER 仍然明显低于六分类的情况，这进一步说明类别平衡及消除罕见类的干扰的重要性。

此外，三分类问题中 LDA 效果优于 RF 与 SVM，与六分类问题恰好相反，我们认为，这可能与城市各种族人口占比变量对于种族判别结果呈现近似线性关系有关，因此，我们进一步进行去除五个城市各种族人口占比特征的实验进行验证。结果展示在**表 5** 中，可以看出，此时 RF 的效果明显优于 LDA，此外，相比于包含种族比例变量时，三者 APER 均大幅提高，也说明了城市环境特征对于 race 判别的重要性。

表 5：两情形下三模型表现

APER	BWH 三分类		BWH 三分类+去除城市种族比例特征	
	训练集	测试集	训练集	测试集
LDA	0.3009	**0.2775**	0.4010	0.3942
RF	0.0094	0.3266	**0.0015**	**0.3790**
SVM	**0.2248**	**0.2843**	0.3256	0.4179

去除城市种族比例特征后的 RF 模型各变量重要性展示于**图 8** 中，可以发现，与**图 6** 所示的情形一致，高中学历率、贫困率与收入等城市特征对判别结果仍起到了十分重要的作用，此外，年龄因素也仍是区分三类的主要依据之一。

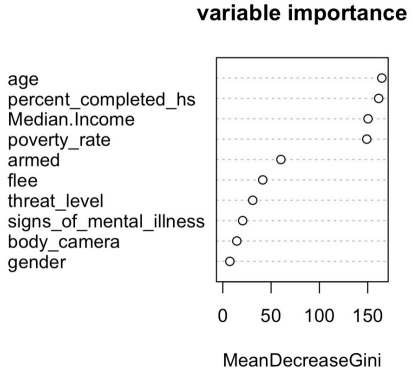

图 8：去除城市种族比例特征的三分类 RF 模型各变量重要性

3.3 典型相关分析

在 PCA 部分的结果中，我们将前两主成分分别解释为代表城市经济文化特征的整体性因素与代表特定案件对象的个体性因素，这种划分在 RF 模型变量重要性排序中也有所体现。这提示我们可以将所有 16 个变量拆分为两类变量进行典型相关分析，以 CCA 的方式分析城市类变量与个体案件类变量之间相关性最大的方向。

首先给出我们的变量分组方式：**表 1** 中的第一列变量归为个体案件组 Event Variables，第二列归为城市特征组 City Variables。与 **3.2.3** 中相似，考虑到 City 组中五个城市种族比例特征对 Event 组中的种族特征存在较为明显的关联，我们对包含该五个变量与不包含这些变量分别分析，分别记为“race”和“no race”。两组变量间的相关性热力图展示在**图 9** 中。

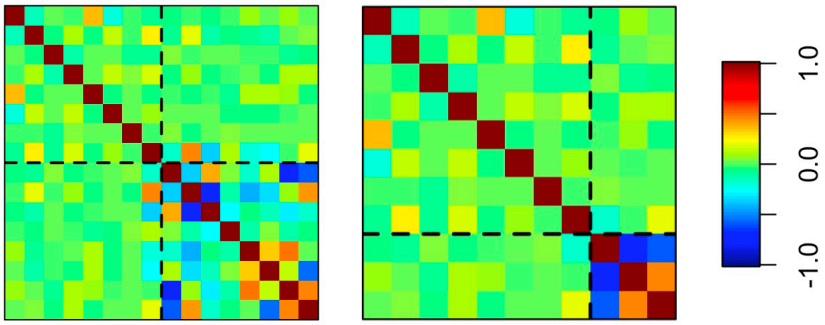

图 9：两组变量间相关性热力图（左：全部变量；右：去除城市种族比例特征）

观察**图 9** 可以发现，在 City 组中，各变量之间的正负相关性均比较显著，而对于 Event 组，变量之间相关性很弱，所携武器与威胁程度、年龄与种族是两组具有相对较明显相关性的变量。此外，正如前所述，race 与城市种族比例特征之间相关性较强。

表 6：典型相关分析结果

Event	xcoef*100		City	ycoef*100		canonical correlation	
Variable	race	no race	Variable	race	no race	race	no race
armed	-0.214	-0.164	%poverty	0.403	2.600	1 **0.5105**	**0.3290**
age	-0.256	-0.197	%white	-0.626	-	2 0.1816	0.1661
gender	-0.223	-0.278	%black	1.260	-	3 0.1548	0.0936
mental_ill	0.124	0.296	%native	-0.117	-	4 0.1314	-
threat	0.288	0.676	%asian	0.238	-	5 0.0924	-
flee	0.004	0.035	%hispanic	1.071	-	6 0.0579	-
camera	0.137	0.261	mid.Income	0.464	2.822	7 0.0434	-
race	-2.120	-2.029	%highschl	0.175	-1.066	8 0.0130	-

两种情形下的 CCA 结果如**表 6** 所示。观察典型相关性数值，均在第一维之后出现明显断层，说明两组间的关联主要通过第一典型相关变量产生，各变量系数于**表 6** 中列出。值得注

意的是，无论是否包含城市种族比例变量，Event 组中 race 的系数都起到了主导地位，明显高于其他变量系数，而对于 City 组的各个变量，当城市种族比例变量存在时，以黑人占比和西班牙裔占比占主要，而当去除城市种族比例变量时，经济因素与教育因素均起到了较重要的作用，其中经济因素的作用相对更大。

以上结果表明，race 是联系 Event 组与 City 组的主要变量，进一步印证了警察执法过程中的种族偏见与城市整体经济文化环境有着重要的联系，是一个整体现象而非个体行为。而在诸多城市环境特征中，少数族裔人口占比、经济发展水平与居民受教育程度三大类因素的作用依次递减，亦即文化传统、经济与教育的因素作用递减。这在一定程度上反映了美国的种族偏见主要来源于整体文化传统，在短时间内较难得到根本性的改善。

4 讨论与改进

本研究采用三种不同的统计分析方法研究警察暴力执法事件中种族偏见现象的相关因素及可能缘由。

根据 PCA，我们将变量提取为前两主成分进行降维可视化，并给出前两主成分的实际含义：表征案件所在城市经济社会环境的整体因素，与表征特定案件本身性质及个体对象的个体因素，这为后来的 CCA 提供了思路。但降维后的数据对种族标签的分散不甚理想，这一方面说明数据难以被低维较好表示，另一方面，我们在未来将尝试 MDS，t-SNE 等可能更好的降维可视化方式。

对于判别分析部分，我们以种族作为类别标签，采用 LDA，QDA，RF，SVM 四种方式进行判别，四种模型在不同情形下有不同的表现，整体而言，RF 在测试集表现最优，但有明显的过拟合现象，值得一提的是，能力较弱的线性判别器 LDA 在本任务表现很好，在一些情况下甚至超过了 RF，一定程度上体现了本任务变量与标签之间的线性性质。然而，不论何种方法，准确率都未达到 0.75 以上，一方面，在 armed 等类别型变量转化为数值型变量的方式有待改进，另一方面，有待我们进一步找寻更适合于本任务数据、能力更强的分类器。此外，在模型之外，特征的缺乏也是效果难以突破的重要因素，进一步的数据丰富及特征工程对本任务十分关键。

对于 CCA 部分，我们按照 PCA 得到的城市组与事件组将变量分为两组，在两种情形下进行典型相关分析，得出社会文化传统的整体性因素与个体种族有较强关联，侧面印证了由来已久的美国种族歧视问题较难短时间解决。但至此，所有的分析均停留在相关性分析层面，未能发掘变量之间的因果关系，未来可以引入因果分析方式，对警察执法所反映出来的种族偏见问题的缘由进行更为可靠深入的讨论。

5 附录

数据集来源于 Kaggle 网站，网址：https://www.kaggle.com/kwullum/fatal-police-shootings-in-the-us。以下列出所有代码，出于个人习惯，数据预处理及 EDA 部分使用 Python 语言，之后的统计学分析部分使用 R 语言，代码如下：

```
# Python 部分

import numpy as np
import pandas as pd
import seaborn as sns
from sklearn import linear_model, metrics
%matplotlib inline
import matplotlib
import matplotlib.pyplot as plt

# data preprocessing
df = pd.read_csv("PoliceKillingsUS.csv", encoding="windows-1252")
df.head()
df = df.rename(columns={"city": "City"})
df.drop(["id", "name", "manner_of_death"], axis=1, inplace=True)
df.age.fillna(value=df.age.mean(), inplace=True)
# Dealing with missing AGE values. Set them to mean of all ages.
df.age = df.age.astype(int)
df.dropna(subset=["race"], inplace=True)
# Deleting rows with missing values for race
df.drop(df.index[2363:], inplace=True)
conditions = [df["race"]=="A", df["race"]=="W", df["race"]=="H",
df["race"]=="B", df["race"]=="N", df["race"]=="O"]
numbers = [14674252, 223553265, 50477594, 38929319, 2932248, 22579629]
df["total_population"] = np.select(conditions, numbers, default="zero")
df.head()

# EDA
# Race
plt.figure(figsize=(6,5))
sns.countplot(data=df, x="race")
plt.grid(alpha = 0.2)
plt.title("Total Number of People Killed, By Race", fontsize=17)
plt.savefig('total_race.png', dpi = 200)

races = ["A", "W", "H", "B", "N", "O"]
killed_per_race = []
for i in races:
    i_killings = df.race.loc[(df.race==i)].count()
    killed_per_race.append(i_killings)
print (killed_per_race)

prop_killed_per_race = []
for i in races:
    if i == "A":
        prop_i_killed = 10000 * killed_per_race[0]/14674252.0
        print (prop_i_killed)
    elif i == "W":
        prop_i_killed = 10000 * killed_per_race[1]/223553265.0
        print (prop_i_killed)
    elif i == "H":
```

```
        prop_i_killed = 10000 * killed_per_race[2]/50477594.0
        print (prop_i_killed)
    elif i == "B":
        prop_i_killed = 10000 * killed_per_race[3]/38929319.0
        print (prop_i_killed)
    elif i == "N":
        prop_i_killed = 10000 * killed_per_race[4]/2932248.0
        print (prop_i_killed)
    else:
        prop_i_killed = 10000 * killed_per_race[5]/22579629.0
        print (prop_i_killed)
    prop_killed_per_race.append(prop_i_killed)

plt.figure(figsize=(6,5))
plt.title("%% Proportion of People Killed", fontsize=17)
plt.grid(alpha = 0.2)
sns.barplot(x=races, y=prop_killed_per_race)
plt.savefig('prop.png', dpi = 200)

# Gender
female = df[df["gender"] == "F"].gender.count()
male = df[df["gender"] == "M"].gender.count()
perc_male = (male*100)/(male+female)
plt.figure(figsize=(3.5,5))
sns.countplot(data=df, x="gender")
plt.grid(alpha = 0.2)
plt.title("Gender Distribution", fontsize=17)
plt.savefig('gender.png', dpi = 200)

# Age
plt.figure(figsize=(9,5))
age_dist = sns.distplot(df["age"], bins=40)
age_dist.set(xlabel="Age", ylabel="Count")
plt.grid(alpha = 0.2)
plt.title("Age distribution", fontsize=17)
plt.savefig('age.png', dpi = 200)

three_races = df.loc[(df["race"] == "B") | (df["race"] == "W") |
(df["race"] == "H")]
plt.figure(figsize=(9,6))
g = sns.FacetGrid(data=three_races, hue="race", aspect=3, size=4)
g.map(sns.kdeplot, "age", shade=True)
g.add_legend(title="Race")
plt.grid(alpha = 0.2)
plt.title("Age distribution, by race", fontsize=17)
plt.savefig('age-race.png', dpi = 200)

avg_age_w = df.age[(df["race"] == "W")].mean()
avg_age_b = df.age[(df["race"] == "B")].mean()
avg_age_h = df.age[(df["race"] == "H")].mean()
print ("Average age of W is " + str(avg_age_w))
print ("Average age of B is " + str(avg_age_b))
print ("Average age of H is " + str(avg_age_h))

plt.figure(figsize=(12,5))
sns.countplot(data=df, x=df.state)
plt.grid(alpha = 0.2)
plt.title("Number of Police Killings, By State", fontsize=17)
plt.savefig('state.png', dpi = 200)
```

```
# City and State
city = df.City.value_counts(ascending=False)
df_city = df.filter(["City"], axis=1)
df_city["count"] = 1
grouped_city = df_city.groupby("City", as_index=False,sort=False).sum()
grouped_city.sort_index(ascending=False)
grouped_city = grouped_city.sort_values("count", ascending=False).head(8)
plt.figure(figsize=(9,5))
sns.barplot(data=grouped_city, x="City", y="count")
plt.grid(alpha = 0.2)
plt.title("Most Dangerous Cities", fontsize=17)
plt.savefig('city_dang.png', dpi = 200)

# Armed
armed = df.armed.value_counts(ascending=False)
df_armed = df.filter(["armed"], axis=1)
df_armed["count"] = 1
grouped_armed = df_armed.groupby("armed", as_index=False,sort=False).sum()
grouped_armed.sort_index(ascending=False)
grouped_armed = grouped_armed.sort_values("count", ascending=False).head(8)
plt.figure(figsize=(10,5))
sns.barplot(data=grouped_armed, x="armed", y="count")
plt.grid(alpha = 0.2)
plt.title("Most Common Ways of Being Armed", fontsize=17)
plt.savefig('armed.png', dpi = 200)

# Merge City Info
income = pd.read_csv("MedianHouseholdIncome2015.csv", encoding="windows-
1252")
income["City"].replace(["city", "CDP", "town"], "", regex=True,
inplace=True)
income["city"] = income["City"] + ", " + income["Geographic Area"]
income.drop(["Geographic Area", "City"], axis=1, inplace=True)
poverty = pd.read_csv("PercentagePeopleBelowPovertyLevel.csv",
encoding="windows-1252")
poverty["City"].replace(["city", "CDP", "town"], "", regex=True,
inplace=True)
poverty["city"] = poverty["City"] + ", " + poverty["Geographic Area"]
poverty.drop(["Geographic Area", "City"], axis=1, inplace=True)
race = pd.read_csv("ShareRaceByCity.csv", encoding="windows-1252")
race["City"].replace(["city", "CDP", "town"], "", regex=True, inplace=True)
race["city"] = race["City"] + ", " + race["Geographic area"]
race.drop(["Geographic area", "City"], axis=1, inplace=True)
highschool = pd.read_csv("PercentOver25CompletedHighSchool.csv",
encoding="windows-1252")
highschool["City"].replace(["city", "CDP", "town"], "", regex=True,
inplace=True)
highschool["city"] = highschool["City"] + ", " + highschool["Geographic
Area"]
highschool.drop(["Geographic Area", "City"], axis=1, inplace=True)

df["city"] = df["City"] + " , " + df["state"]
merge1 = pd.merge(poverty, race, on="city", how="outer")
merge2 = pd.merge(merge1, income, on="city", how="outer")
merge3 = pd.merge(merge2, highschool, on="city", how="outer")
data = pd.merge(df, merge3, on="city", how="outer")
data.dropna(inplace=True)
```

```
data[["Median Income", "poverty_rate", "share_white", "share_black",
"share_native_american", "share_asian", "share_hispanic",
"percent_completed_hs"]] = data[["Median Income", "poverty_rate",
"share_white", "share_black", "share_native_american", "share_asian",
"share_hispanic", "percent_completed_hs"]].replace("(X)", np.NaN)
data[["Median Income", "poverty_rate", "share_white", "share_black",
"share_native_american", "share_asian", "share_hispanic",
"percent_completed_hs"]] = data[["Median Income", "poverty_rate",
"share_white", "share_black", "share_native_american", "share_asian",
"share_hispanic", "percent_completed_hs"]].replace("-", np.NaN)
data[["Median Income", "poverty_rate", "share_white", "share_black",
"share_native_american", "share_asian", "share_hispanic",
"percent_completed_hs"]] = data[["Median Income", "poverty_rate",
"share_white", "share_black", "share_native_american", "share_asian",
"share_hispanic", "percent_completed_hs"]].astype(float)

data.dropna(inplace=True)

# Converting neccesary columns to floats
data["poverty_rate"] = data["poverty_rate"].astype(float)
data["share_white"] = data["share_white"].astype(float)
data["share_black"] = data["share_black"].astype(float)
data["share_native_american"] = data["share_native_american"].astype(float)
data["share_asian"] = data["share_asian"].astype(float)
data["share_hispanic"] = data["share_hispanic"].astype(float)
data["percent_completed_hs"] = data["percent_completed_hs"].astype(float)
data["Median Income"] = data["Median Income"].astype(int)
data.head()

# Dealing With Categorical Variables
data["signs_of_mental_illness"] =
data["signs_of_mental_illness"].astype(int)
data["body_camera"] = data["body_camera"].astype(int)
print(np.max(data['body_camera']))

from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
le.fit(["armed", "race", "gender", "city", "state", "threat_level",
"flee"])
data_log = data.apply(LabelEncoder().fit_transform)
X = data_log
y = data_log["race"]
data_log.head()
data_log.to_csv("transform_data.csv")



# R 语言部分

library(CCA)
library(CCP)
library(MASS)
library(ggplot2)
library(randomForest)
library(e1071)
library(gg3D)
final_data = read.csv('/Users/monica/Desktop/final_data.csv')

# PCA
```

```
scale_data = scale(final_data[,1:16])
pca = prcomp(scale_data[, c(1:8, 10, 13, 14:16)])
evals<-data.frame(pca$sdev^2)
names(evals)<-"eigen.vals"
evals$component.num<-as.integer(seq(nrow(evals)))
ggplot(evals,aes(x=component.num,y=eigen.vals))+geom_point()
pca$rotation

# 降维可视化
ggplot(data.frame(pca$x[, 1], pca$x[, 2], final_data$Race_str),
       aes(x=pca.x...1.,y=pca.x...2.,
           col=final_data.Race_str,shape = final_data.Race_str)) +
  xlab('pca_1')+ ylab('pca_2')+ geom_point() +
  theme(legend.position='bottom')

# 3D 图
theta=20
phi=30
ggplot(data.frame(pca$x[, 1], pca$x[, 2], pca$x[, 3], final_data$Race_str),
       aes(x=pca.x...1.,y=pca.x...2., z = pca.x...3.,
           col=final_data.Race_str,shape = final_data.Race_str)) +
  axes_3D(theta=theta, phi=phi) + stat_3D(theta=theta, phi=phi) +
  axis_labs_3D(theta=theta, phi=phi, size=3, hjust=c(1,1,1.2,1.2,1.2,1.2),
               vjust=c(-.5,-.5,-.2,-.2,1.2,1.2)) +
  labs_3D(theta=theta, phi=phi, hjust=c(1,0,0), vjust=c(1.5,1,-.2),
          labs=c("pca_1", "pca_2", "pca_3")) +theme_void() +
  theme(legend.position='bottom')

# 判别分析
# 将数据集分为训练集和测试集,比例为 7:3
set.seed(1)
final_data[1:15] = scale(final_data[1:15])
train_sub = sample(nrow(final_data), 7/10*nrow(final_data))
train_data = final_data[train_sub, c(1:15, 17)]
test_data = final_data[-train_sub, c(1:15, 17)]

# LDA
L = lda(Race_str~.,data=final_data[, c(1:15, 17)])
yhat = predict(L, final_data[, c(1:15, 17)])$class
final_data$pred_race = yhat

ggplot(data.frame(pca$x[, 1], pca$x[, 2], final_data$pred_race,
final_data$Race_str),
aes(x=pca.x...1.,y=pca.x...2.,col=final_data.pred_race,shape =
final_data.Race_str)) +
  geom_point() + xlab('pca_1')+ ylab('pca_2')+
theme(legend.position='bottom')

tab = table(pred=final_data$pred_race, true=final_data$Race_str); tab
aper = sum(tab[row(tab)!=col(tab)])/sum(tab); aper

# LDA + CV
L = lda(Race_str~., data=train_data)
pre_ran = predict(L, newdata=test_data)$class
tabcv = table(pred=pre_ran, true=test_data$Race_str);tabcv
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr
pre_train = predict(L, newdata=train_data)$class
tabcv = table(pred=pre_train, true=train_data$Race_str);tabcv
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr
```

```
# QDA
Q = qda(Race_str~., data=train_data)
pre_ran <- predict(Q, newdata=test_data)$class
tabcv = table(pred=pre_ran, true=test_data$Race_str);tabcv
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr
pre_train = predict(Q, newdata=train_data)$class
tabcv = table(pred=pre_train, true=train_data$Race_str);tabcv
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr

# RF
# choose tree number
testerr = c(); trainerr = c()
for (i in 1:10){
  rf = randomForest(Race_str~., data=train_data, ntree = 25 * i)
  pre_ran_test = predict(rf, newdata=test_data)
  tabcv_test = table(pred=pre_ran_test, true=test_data$Race_str)
  cverr_test =
sum(tabcv_test[row(tabcv_test)!=col(tabcv_test)])/sum(tabcv_test)
  testerr = c(testerr, cverr_test)
  pre_ran_train = predict(rf, newdata=train_data)
  tabcv_train = table(pred=pre_ran_train, true=train_data$Race_str)
  cverr_train =
sum(tabcv_train[row(tabcv_train)!=col(tabcv_train)])/sum(tabcv_train)
  trainerr = c(trainerr, cverr_train)
}
testerr; trainerr

tree_num = c(25 * (1:10), 25 * (1:10))
err = c(trainerr, testerr)
type = c(rep('train', 10), rep('test', 10))
ggplot(data.frame(tree_num, err), aes(x=tree_num, y=err, color=type,
shape=type)) + geom_line() + geom_point(size=3)

rf = randomForest(Race_str~., data=train_data, ntree = 175)
varImpPlot(rf, main = "variable importance")
pre_ran <- predict(rf, newdata=test_data)
tabcv = table(pred=pre_ran, true=test_data$Race_str);tabcv
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr
ran_roc <- roc(test_data$Race_str, as.numeric(pre_ran))
plot(ran_roc, print.auc=TRUE, auc.polygon=TRUE, grid=c(0.1, 0.2),grid.col
=c("green", "red"), max.auc.polygon=TRUE, auc.polygon.col="skyblue",
print.thres=TRUE, main='RF ROC Curve')

# SVM
svm = svm(Race_str~., data=train_data, type = 'C',kernel = 'radial')
pre_ran_test <- predict(svm, newdata=test_data)
tabcv_test = table(pred=pre_ran_test, true=test_data$Race_str)
cverr_test =
sum(tabcv_test[row(tabcv_test)!=col(tabcv_test)])/sum(tabcv_test);
cverr_test
pre_ran_train <- predict(svm, newdata=train_data)
tabcv_train = table(pred=pre_ran_train, true=train_data$Race_str)
cverr_train =
sum(tabcv_train[row(tabcv_train)!=col(tabcv_train)])/sum(tabcv_train);cverr
_train
ran_roc <- roc(test_data$Race_str, as.numeric(pre_ran_test))
plot(ran_roc, print.auc=TRUE, auc.polygon=TRUE, grid=c(0.1, 0.2), grid.col
```

```
=c("green", "red"), max.auc.polygon=TRUE,auc.polygon.col="skyblue",
print.thres=TRUE, main='SVM ROC Curve')

# 三分类
final_data = read.csv('/Users/monica/Desktop/final_data.csv')
new_data = final_data[final_data$Race_str %in% c('B', 'H', 'W'),]
new_data$Race_str = factor(new_data$Race_str)
new_data[1:15] = scale(new_data[1:15])
new_train_sub = sample(nrow(new_data), 7/10*nrow(new_data))
new_train_data_1 = new_data[new_train_sub, c(1:15, 17)]
new_test_data_1 = new_data[-new_train_sub,c(1:15, 17)]
new_train_data_2 = new_data[new_train_sub, c(1:8, 14:15, 17)]
new_test_data_2 = new_data[-new_train_sub,c(1:8, 14:15, 17)]

# with race% info
# LDA
L = lda(Race_str~., data=new_train_data_1)
pre_ran_test <- predict(L, newdata=new_test_data_1)$class
tabcv = table(pred=pre_ran_test, true=new_test_data_1$Race_str)
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr
pre_ran_train <- predict(L, newdata=new_train_data_1)$class
tabcv = table(pred=pre_ran_train, true=new_train_data_1$Race_str)
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr

# RF
rf = randomForest(Race_str~., data=new_train_data_1, ntree = 175)
varImpPlot(rf, main = "variable importance")
pre_ran_test <- predict(rf, newdata=new_test_data_1)
tabcv = table(pred=pre_ran_test, true=new_test_data_1$Race_str)
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr
pre_ran_train <- predict(rf, newdata=new_train_data_1)
tabcv = table(pred=pre_ran_train, true=new_train_data_1$Race_str)
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr

# SVM
svm = svm(Race_str~., data=new_train_data_1, type = 'C',kernel = 'radial')
pre_ran_test <- predict(svm, newdata=new_test_data_1)
tabcv = table(pred=pre_ran_test, true=new_test_data_1$Race_str)
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr
pre_ran_train <- predict(svm, newdata=new_train_data_1)
tabcv = table(pred=pre_ran_train, true=new_train_data_1$Race_str)
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr

# without race% info
# LDA
L = lda(Race_str~., data=new_train_data_2)
pre_ran_test <- predict(L, newdata=new_test_data_2)$class
tabcv = table(pred=pre_ran_test, true=new_test_data_2$Race_str)
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr
pre_ran_train <- predict(L, newdata=new_train_data_2)$class
tabcv = table(pred=pre_ran_train, true=new_train_data_2$Race_str)
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr

# RF
rf = randomForest(Race_str~., data=new_train_data_2, ntree = 175)
varImpPlot(rf, main = "variable importance")
pre_ran_test <- predict(rf, newdata=new_test_data_2)
tabcv = table(pred=pre_ran_test, true=new_test_data_2$Race_str)
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr
```

```
pre_ran_train <- predict(rf, newdata=new_train_data_2)
tabcv = table(pred=pre_ran_train, true=new_train_data_2$Race_str)
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr

# SVM
svm = svm(Race_str~., data=new_train_data_2, type = 'C',kernel = 'radial')
pre_ran_test <- predict(svm, newdata=new_test_data_2)
tabcv = table(pred=pre_ran_test, true=new_test_data_2$Race_str)
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr
pre_ran_train <- predict(svm, newdata=new_train_data_2)
tabcv = table(pred=pre_ran_train, true=new_train_data_2$Race_str)
cverr = sum(tabcv[row(tabcv)!=col(tabcv)])/sum(tabcv); cverr

# CCA
event = scale_data[, c(1:7, 16)]
city = scale_data[, c(8, 14, 15)]
matcor(event,city)
img.matcor(matcor(event,city))
cca<-cancor(event, city)
rho <- cca$co; rho
cca$xcoef; cca
```