对 Spotify 数据集的分析

目的

目的是更好地预测一首歌是否被此人喜欢。有 2 个目标：1. 作出更准确的分类，为此，用 LDA,QDA 进行分类 2. 使分类标准更加易于解释。为此，试图用 PCA 和因子分析降维可视化。由于这些歌曲没有类别信息，做一下聚类分析和因子分析，用 k-means 方法，看一下做出来的类别是否与喜好[1]有明显的关系。

数据的含义

- acousticness[2]，未被电子放大，接近 1 表示没有放大。
- energy: 无需解释，同样在 0 到 1 之间。
- instrumentalness：越接近 1，表示人声越少。
- key：因子变量，c# 这样。
- liveness：越接近 1，表示现场版的可能性越大，如果 >0.8 则很有可能是现场版。
- loudness：用 db 刻画。
- mode：因子变量，大调还是小调。
- speechiness：越接近 1 表示音乐越少，说得越多，类似于 talk show 和演讲等。0.33-0.66 就是有说又有音乐，比如 rap。
- tempo：BPM, 每分钟有几小节。
- time_signatures：每小节有几拍。
- valence：悲哀程度，如果比较高，那么比较积极欢快，否则消极 (抑郁、生气、悲伤)。

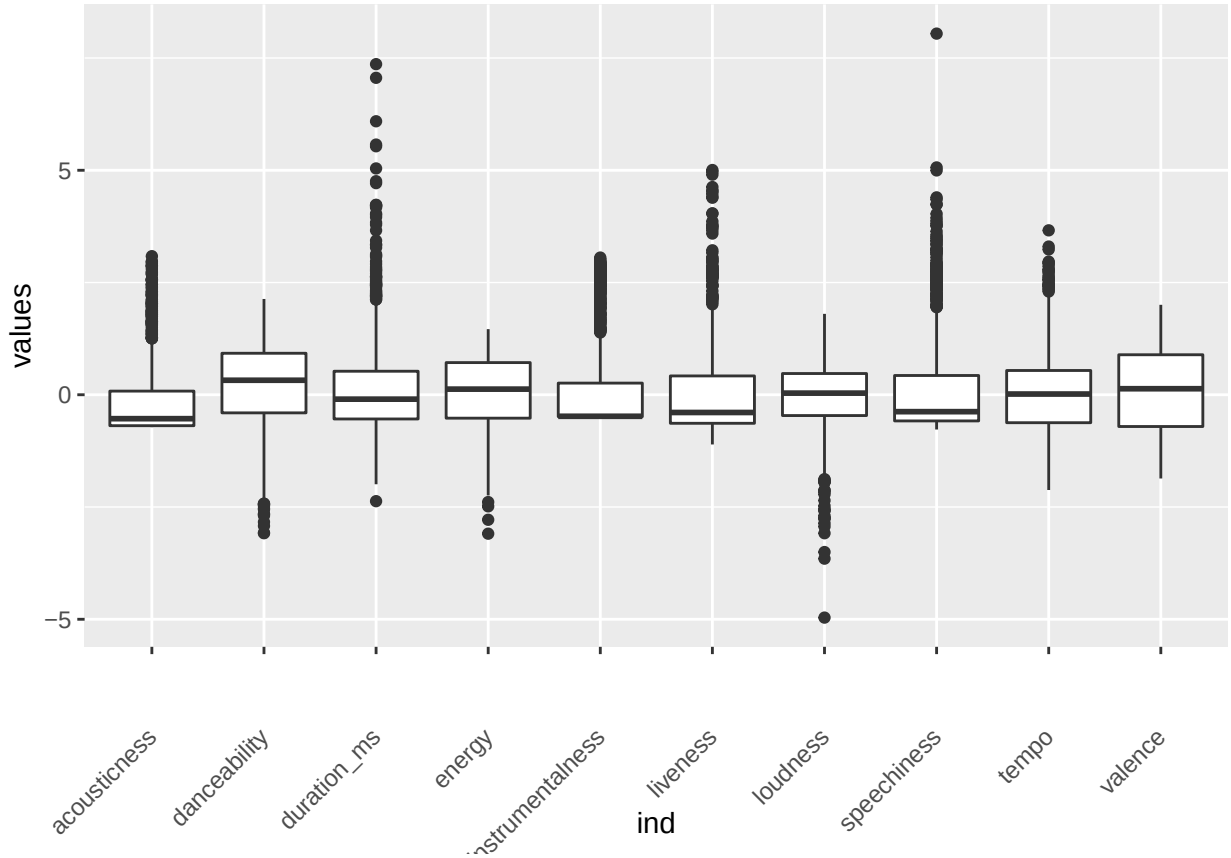
EDA

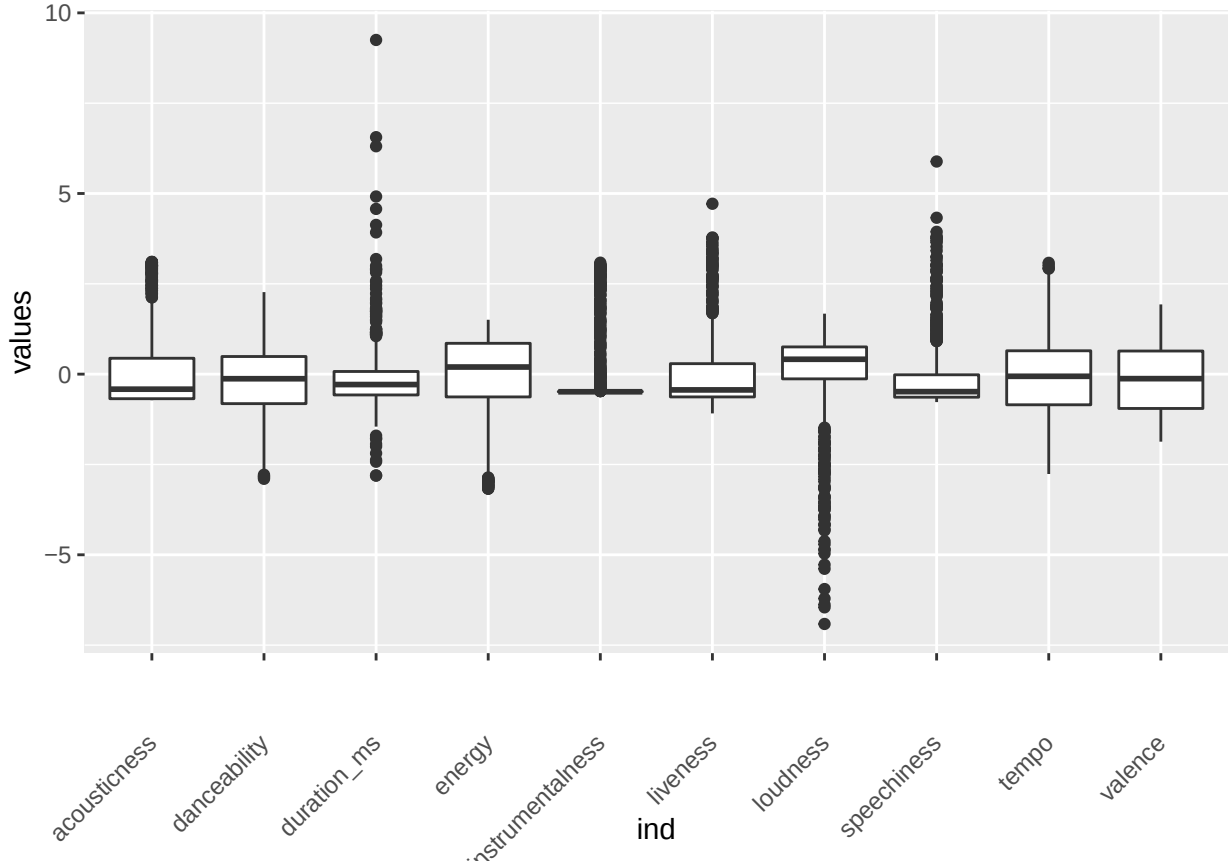
由于歌手众多 (1600 多个不同歌手)，作为因子，意义很小，而运算量却非常大，非常明显地影响运行速度，因此在后面单独分析常见歌手。其它类别变量转换为因子。对数据做标准化，因为尺度不一，比如，duration_ms 远大于其它量。

[1]为了叙述方便，把这个人称为 A

[2]acoustic: Of a musical instrument, gramophone, etc.: not electrically amplified.

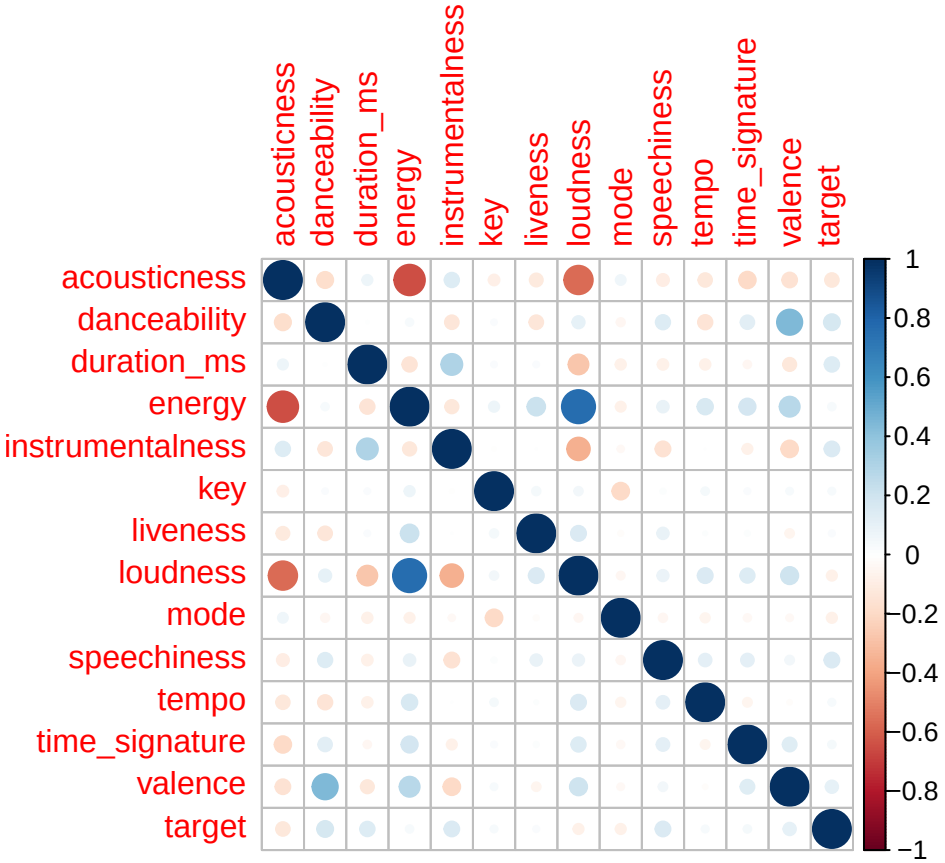
数据的总体描述


values
5
0
−5
acousticness
danceability
duration_ms
energy
instrumentalness
liveness
loudness
speechiness
tempo
valence
ind


values
10
5
0
−5
acousticness
danceability
duration_ms
energy
instrumentalness
liveness
loudness
speechiness
tempo
valence
ind

观察 like 与 dislike 前后的偏移，发现的明显区别是：danceability 的下移，从均值 >0 变为均值 <0，说明 A 更喜欢比较律动的歌曲，duration_ms 也有所下移，说明更厌恶时长偏短一些的，loudness 有明显上升，说明不喜欢更吵的歌曲，valence 下移，说明 A 不喜欢偏消极的歌曲。以上都只是倾向，并不是准则。我们据此得到一个更重要变量的索引。tempo 和 liveness 并不重要，影响很小。

变量的相关性



由图，多数变量之间的相关性很弱，比较强的相关性主要存在于 energy,loudness,acousticness 之间，以及 valence 与 danceability，这符合常识。但总体来说，数据的相关性偏弱，使人怀疑做 PCA、因子分析、LDA 这些基于线性的方法效果可能不会很好。

喜欢和不喜欢的歌手

```
##                   target
## artist              0  1
##   *NSYNC            8  0
##   Backstreet Boys  10  0
##   Big Time Rush     8  0
##   Crystal Castles   0  9
```

```
##   Demi Lovato      8  0
##   Disclosure       0 12
##   Drake            3 13
##   Fall Out Boy     8  0
##   FIDLAR           0  9
##   Future           2  6
##   Kanye West       0  8
##   Kina Grannis     8  0
##   Michael Jackson  8  0
##   Rick Ross        9  4
##   Skrillex         8  0
##   WALK THE MOON   10  0
```

通过尝试性地调整 top_thresh，希望能控制 artist 的数量，确定 thresh>=8 次以上，这样有 16 个歌手[3]。

虽然不一定能全面反映 A 的喜好，但也能得到他最喜欢和最不喜欢的一些 artist。- 最不喜欢的：*NSYNC,Backstreet Boys,Big Time Rush,Demi Lovato,Fall Out Boy,Kina Grannis,Skrillex,WALK THE MOON - 最喜欢的：Crystal Castles,Disclosure,FIDLAR,Kanye West,Michael Jackson 这两个名单的特点是，全不喜欢或者全是喜欢。Drake 出现次数为最高，喜欢和不喜欢分别是 10 和 3，还是偏向于喜欢的，Future 也如此。Rick Ross 则相反，分别是 9 和 4，偏向于不喜欢。

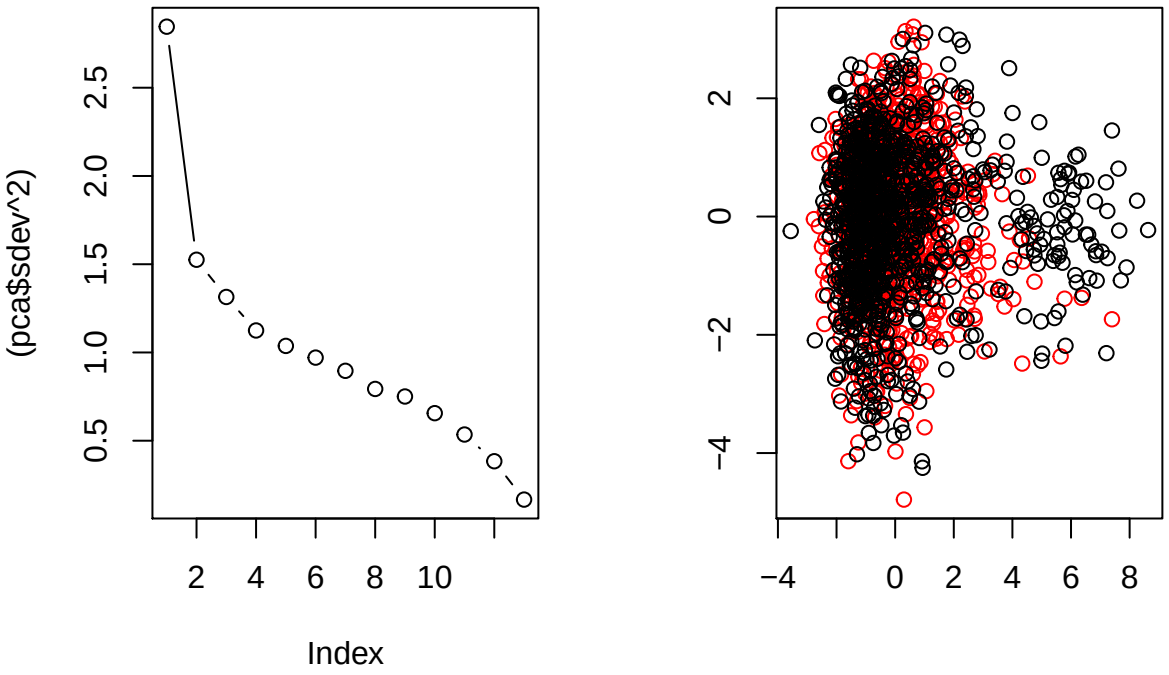
尝试降维可视化

由于变量个数太多，不易解释，考虑降维。

PCA

```
## [1] 0.7472668
```

[3]由于 thresh 固定，CHVRCHES(喜欢),The Chainsmokers(不喜欢),Young Thug(喜欢) 出现 7 次，也算高频，但没有入选



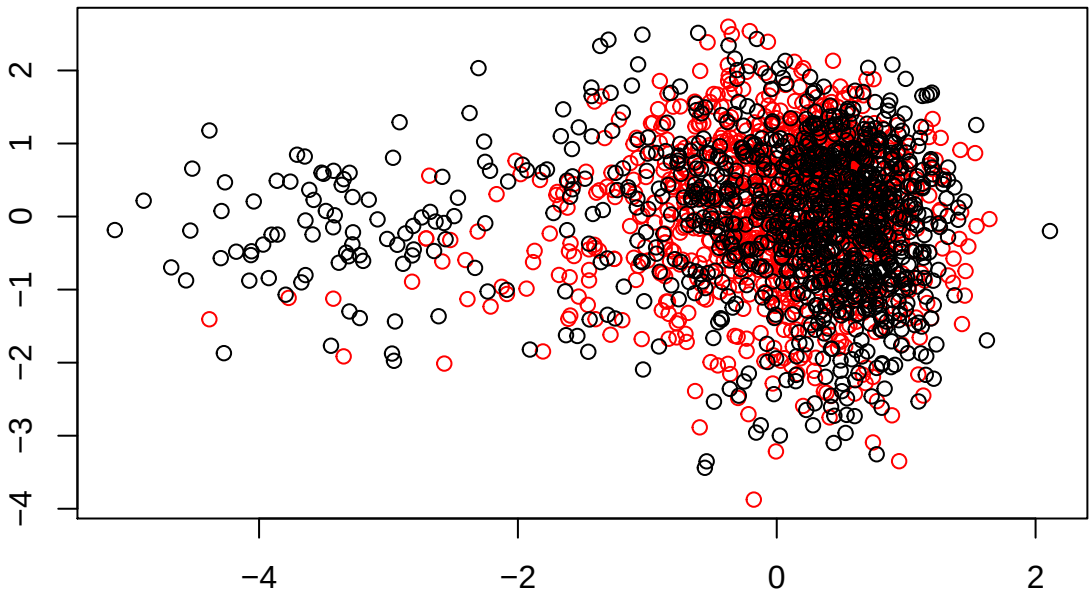
PCA 的降维效果不怎么好，从这幅图来看，至少应该保留 7 个主成分，而且就算是 7 个，也只解释了 74.7% 的方差，二维可视化可以不用想了，不过还是做做看，事实上也的确如此，混杂在一起，无法分开，不过也还是有一些特点，在 PC1 和 PC2 比较大的时候 (靠近图的右上角)，不喜欢的比例高很多。这符合常识，被喜欢的一般不会太极端。

如果 drop 一些变量，对方差解释度会提高，这在预期之内，但对分类无济于事。

因子分析　做因子分析，同样在只有 2 个变量时，解释度很低

```
## [1] 0.3362497
```

解释度如此低，我猜测与因子有关，在去除因子变量后 (6,9,12)，也没有提高多少 (33.6%% 从 42.7%)，与相关性比较差有关。



有着类似的效果，因子得分与类别完全混在一起。去除变量后也类似，不再多说。

分类

LDA

10 折交叉验证的结果为如下，错误率很高

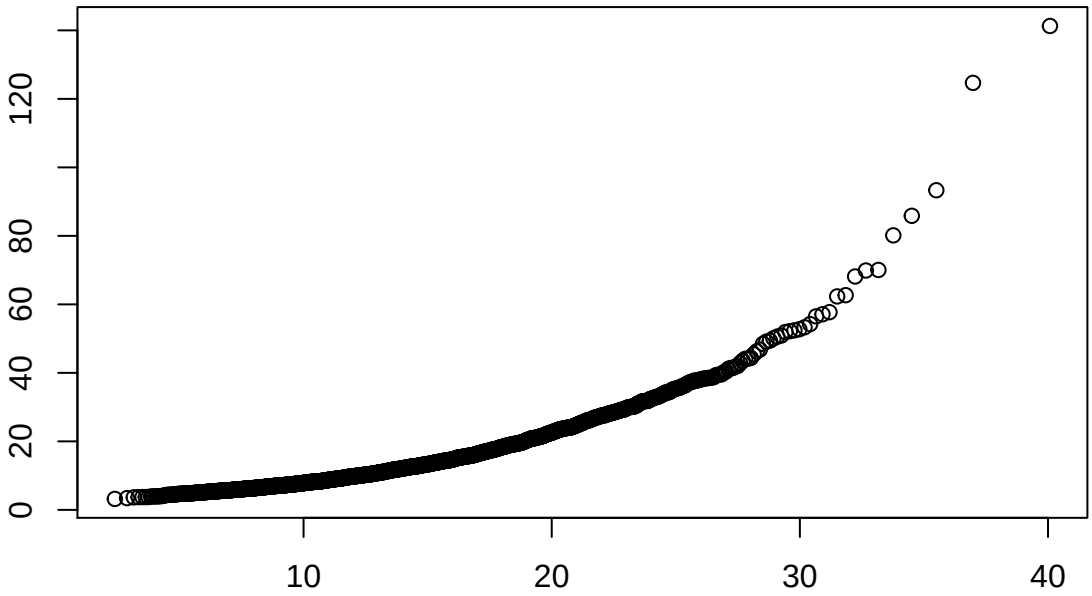
```
## [1] 0.3465322
```

QDA

用 qda，错误率有所降低，不过还是很高

```
## [1] 0.3004335
```

正态性如何？



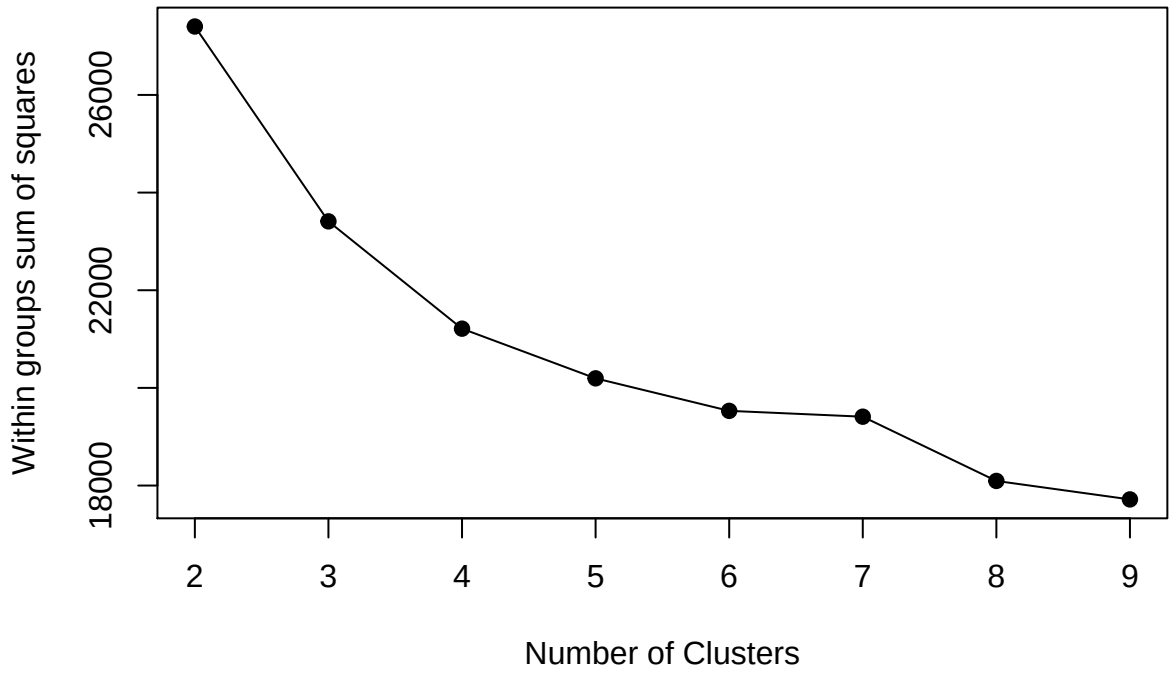
并不接近直线，完全是曲线，正态性是不满足的，何况这只是一个并不全面的检验。用 R 提供的一个多元正态性的检验。p 值很高，与正态性完全不沾边。而 QDA 对正态性是敏感的，但它效果能好于 LDA，原因大概是：LDA 太差了。也说明了线性判别在这里并不是好的做法。

```
##
##  Energy test of multivariate normality: (Specify R > 0 for MC test)
##
## data:  x, sample size 2017, dimension 14, replicates 0
## E-statistic = 596188, p-value = NA
```

聚类

聚类是无监督的方法，一般我认为它的效果是不如 LDA 的。但是由于线性方法在这里效果并不很好，它是非线性的，所以我觉得它可能会有更好的表现。

分为几类



由 elbow method，分成 4 类是合适的。

2 类

先看看 2 类的情况，看看能否有助于 like/dislike 的分类。

```
##
##        0    1
##    1 492 534
##    2 505 486
```

参考价值很小，因为第 2 类的数目比第 1 类要多得多，无论是喜欢还是不喜欢。如果尝试 drop 一些变量，也同样如此

```
##
##        0    1
##    1 852 937
##    2 145  83
```

4 类

划分为 4 类

```
##
```

```
##       0   1
##   1 741 589
##   2  72 178
##   3 137  65
##   4  47 188
```

从分类的角度，没什么收获，因为没有表现出非常显著、足以决定分类的差异。第 1 类和第 4 类的比例对 0,1 差不太多，2 大致为 1:2，也不算悬殊，第 3 组有比较明显的差别，约 3:1，但也不是决定性的。

与歌手对比

关于分类的合理性，在认知中，同一歌手的风格应该相似

```
##
##     *NSYNC Backstreet Boys Big Time Rush Crystal Castles Demi Lovato Disclosure
##   1      8              10             8               9           8         12
##   2      0               0             0               0           0          0
##
##     Drake Fall Out Boy FIDLAR Future Kanye West Kina Grannis Michael Jackson
##   1    16            8      9      8          8            5               8
##   2     0            0      0      0          0            3               0
##
##     Rick Ross Skrillex WALK THE MOON
##   1        13        7            10
##   2         0        1             0
```

在 2 分类的结果下，几乎全部判别为同一种类型，不过在前面可以看到，本来第 1 类的数目远小于第 2 类的数目。要判断聚类，不妨选类数最多的，我们期待的效果是，虽然有 5 类，但是能集中在 1-2 类中，以下是 5 类的情况。

```
##
##     *NSYNC Backstreet Boys Big Time Rush Crystal Castles Demi Lovato Disclosure
##   1      0               0             0               0           1          3
##   2      8              10             8               6           7          6
##   3      0               0             0               3           0          3
##   4      0               0             0               0           0          0
##
##     Drake Fall Out Boy FIDLAR Future Kanye West Kina Grannis Michael Jackson
##   1    11            0      0      4          3            0               0
##   2     5            8      7      4          5            6               8
##   3     0            0      2      0          0            0               0
##   4     0            0      0      0          0            2               0
```

```
##
##     Rick Ross Skrillex WALK THE MOON
##   1         4        0             0
##   2         9        6            10
##   3         0        1             0
##   4         0        1             0
```

在这一点上，确实满足了期待。

总结

线性方法在这里效果都不好。分类预测效果很差。去网上搜更多的结果，效果最好的是 random forest(80%-83% 的正确率) 之类的非线性方法。LDA,QDA 普遍只有 68%-73% 的正确率，其它的线性方法比如 logistic regression，效果也不佳。说明线性方法是很有局限性的。如果 LDA 解决不了，也就不能指望 PCA 和因子分析能够提高分类效果，更好地解释结果。它们都是线性方法。一度试图用聚类分析解决，因为它不是线性方法，但效果更差，无监督还是不能与有监督的相比。不过聚类并不是完全没有意义，至少高频的 artist 的类型主要集中在 1 类或者 2 类中 (5 类时)。

这个过程中让我苦恼的就是 factor 类型变量，也就是类别变量。之前学习这些方法时，忽略了这样的变量的存在。现在的处理是转换成数值，这个处理很糟糕，因为它们并没有数值大小上的关系。想法是找到合适的变换，还需要进一步看看有没有这方面的结果。