数据挖掘大作业最终报告

——选题《阿里音乐流行趋势预测》

余梦巧 2120171089

张逸恒 2120171100

曹倩雯 3120170497

简介

目前，音乐已成为人们生活娱乐不可或缺的一部分。而在海量的音乐中，如何发掘出有价值的信息是一个很有趣的问题。基于此，我们选择对音乐的流行趋势做预测。本次项目的数据来源于天池大赛比赛中的阿里云流行趋势预测大赛。下面对本项目做介绍。

经过多年的发展与沉淀，目前阿里音乐拥有数百万的曲库资源，每天千万的用户活跃在平台上，拥有数亿人次的用户试听、收藏等行为。在原创艺人和作品方面，更是拥有数万的独立音乐人，每月上传上万个原创作品，形成超过几十万首曲目的原创作品库，如此庞大的数据资源库对于音乐流行趋势的把握有着极为重要的指引作用。

本项目以阿里音乐用户的历史播放数据为基础，期望对阿里音乐平台上每个阶段艺人的试听量的预测，挖掘出即将成为潮流的艺人，从而实现对一个时间段内音乐流行趋势的准确把控。

数据介绍

数据集地址：
https://pan.baidu.com/s/1XaAn0NLAJRLelIOwRAGvLg

本项目总共包含两个数据集，分别是
mars_tianchi_songs.csv 和 mars_tianchi_user_actions.csv。

数据集包含了抽样的歌曲艺人数据，以及和这些艺人相关的 6 个月内（20150301-20150830）的用户行为历史记录。包含 100 个艺人，超过 26000 首歌曲，15000000 条用户行为记录。数据表如下，每一行表示一条行为记录，所有可能涉及到隐私的数据在获取之前已经进行脱敏处理。行为记录包括用户 ID、歌曲 ID、用户播放的时间（精确到小时）、行为类型（播放、下载、收藏）、记录搜集日期等。

用户行为表（mars_tianchi_user_actions）

列名	类型	说明	示例
user_id	String	用户唯一标识	7063b3d0c075a4d276c5f06f4327cf4a
song_id	String	歌曲唯一标识	effb071415be51f11e845884e67c0f8c
gmt_create	String	用户播放时间（unix 时间戳表示）精确到小时	1426406400
action_type	String	行为类型：1，播放；2，下载，3，收藏	1
Ds	String	记录收集日（分区）	20150315

同时也提供了歌曲艺人的数据表，包括歌曲 ID、艺人 ID、歌曲发行时间（精确到天）、歌曲初始播放次数、歌曲语言、艺人性别或者组合等。

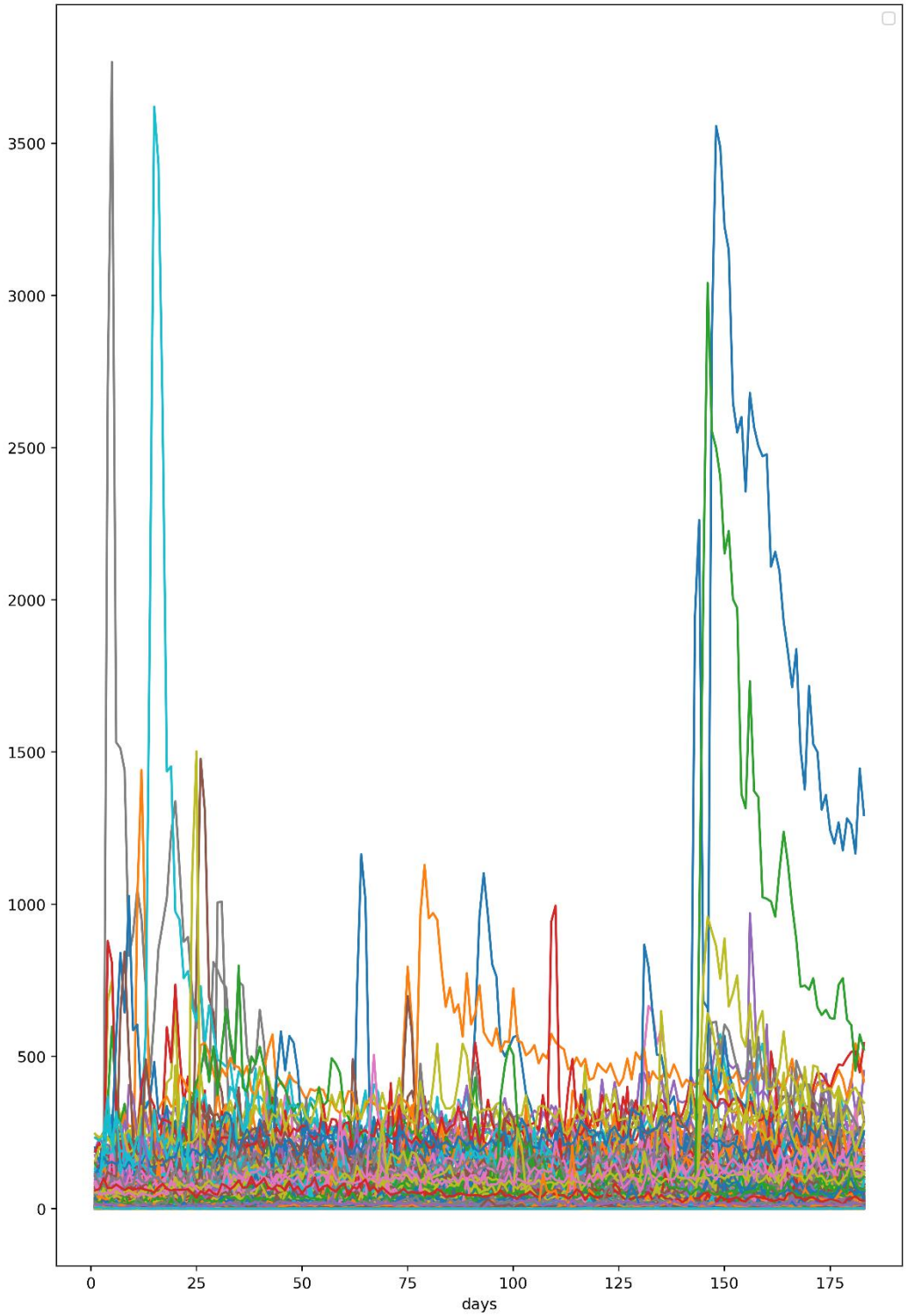
歌曲艺人（mars_tianchi_songs）

列名	类型	说明	示例
song_id	String	歌曲唯一标识	c81f89cf7edd24930641afa2e411b09c
artist_id	String	歌曲所属的艺人 Id	03c6699ea836decbc5c8fc2dbae7bd3b
publish_time	String	歌曲发行时间，精确到天	20150325
song_init_plays	String	歌曲的初始播放数，表明该歌曲的初始热度	0
Language	String	数字表示 1,2,3…	100
Gender	String	1,2,3	1

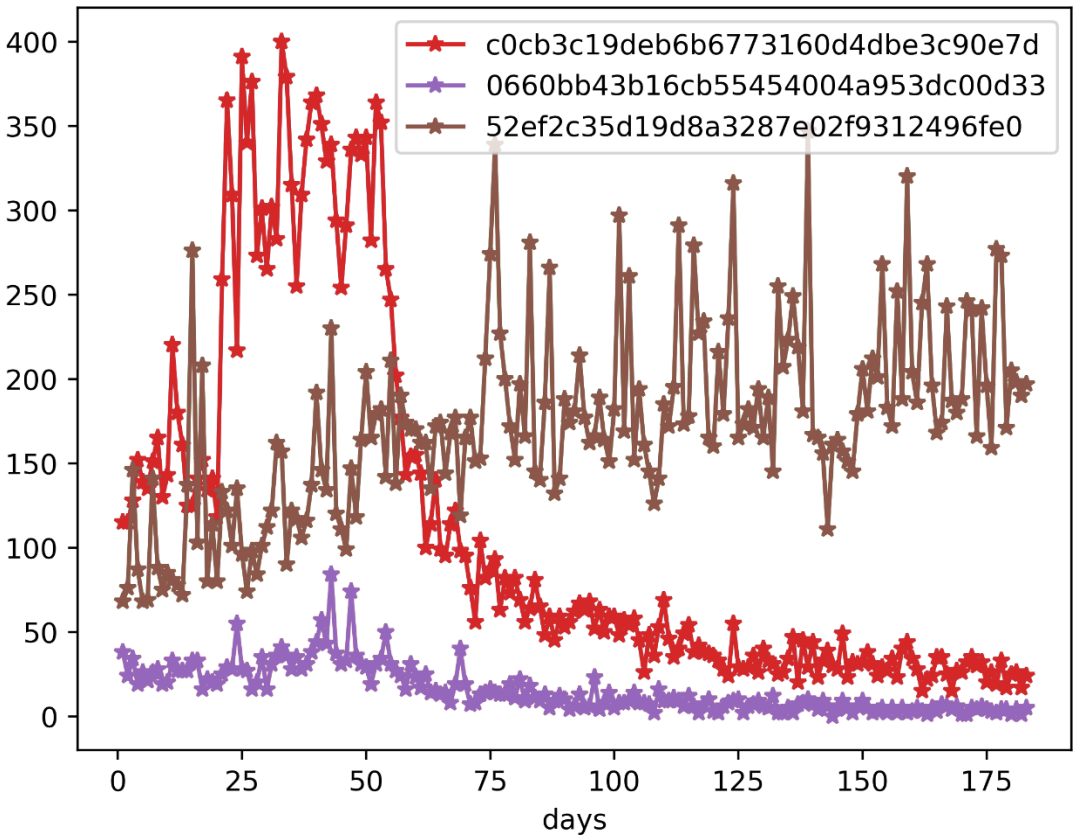
我们所做的就是根据以上两个表，预测在接下来的两个月（20150901-20151030）范围内，每个艺人每天的歌曲播放次数。

项目分析

我们尝试从歌曲特征（下载、收藏、播放等），歌手特征（下载量、播放量、粉丝量），用户特征（每天听歌次数），行为特征（下载对播放的影响）四个方向对给定的数据集做粗略的可视化分析。下面展示了两幅可视化预处理的结果图。实验发现，数据很稀疏，尤其是用户方面的数据。最终，经过权衡，选择使用歌曲特征和用户相关的信息来进行预测。



由此图可以发现，绝大多数的歌曲每天的播放量少于 500，而且有一定的波动。少量几首歌曲非常流行，每天播放量达到 3500 以上。这幅图让我们对歌曲的播放量有了一个直观的印象。为了更清楚地看清每天歌曲播放量的变化趋势，我们挑出了其中三天的歌曲播放量数据做可视化，结果如下图。



由此可以看出，歌曲每天的播放量有一定的波动。有一些歌曲始终播放量不大，且变化平稳，这类歌曲可能我没有足够的宣传且质量一般（紫线）；而有些歌曲一开始播放量很大，而逐渐减少，可能宣传做得很好，而歌曲本身的质量无法保证（红线）；还有一些歌曲播放量在波动中走高，这部分歌曲可以说是得到大众认可的歌曲（棕线）。

技术方案

实现的主要思想是针对歌曲和用户两方面进行建模，模型选择 arima 模型。然后将这两个模型相融合来进行最终的预测。
这里首先介绍 arima 模型。

arima 模型介绍

arima 模型由 Box 与 Jenkins 于上世纪七十年代提出，是一种著名的时间序列预测方法。arima 的含义是单积自回归移动平均过程，其含义为：假设一个随机过程含有 d 个单位根，其经过 d 次差分后可以变换为一个平稳的自回归移动平均过程，则该随机过程称为单积（整）自回归移动平均过程。 换句话解释即为，“假设一个随机过程含有 d 个单位根，其经过 d 次差分后可以变换为一个平稳的自回归移动平均过程，则该随机过程称为单积（整）自回归移动平均过程。”

arima 建模的过程可以分成以下几步：
1. 获取被观测系统时间序列数据；

2. 对数据绘图，观测是否为平稳时间序列；对于非平稳时间序列要先进行 d 阶差分运算，化为平稳时间序列；
3. 经过第二步处理，已经得到平稳时间序列。要对平稳时间序列分别求得其自相关系数 ACF 和偏自相关系数 PACF，通过对自相关图和偏自相关图的分析，得到最佳的阶层 p 和阶数 q
4. 由以上得到的 d、q、pd、q、p，得到 ARIMA 模型。然后开始对得到的模型进行模型检验。

建模过程

建模过程分成两步，分别是对歌曲和用户进行建模。

1. 针对歌曲建模

（1） 针对没收歌曲每天的播放量构建 arima 模型，然后预测后 60 天的歌曲播放量。

具体的实现过程为：

在 python 中进行数据预处理，得到歌曲每天的播放量，实现代码为：

```
def get_song_dayly_play(outfl, actions, songs):
    play_df = actions[actions['action_type'] == 1][['song_id',
'action_type', 'ds']]
    dayly_play_amount = play_df.groupby(['song_id',
'ds'])['action_type'].count().unstack()
    amounts = DataFrame(dayly_play_amount,
index=songs['song_id'])
    amounts.fillna(0).to_csv(outfl, header=None)
    return
```

用 R 语言进行 arima 建模，并对之后 60 天的歌曲播放量做粗预测。实现代码为：

```
input_file <- 'C:\\Users\\Administrator\\Desktop\\DM Project\\mars_song_dayly_play.csv'
output_file <- 'C:\\Users\\Administrator\\Desktop\\DM Project\\song_predict_tmp.csv'
given_day_num <- 183   # 用前 6 个月的数据训练
predict_day_num <- 61   # 对后两个月的行为预测

library(forecast)
origin_data <- read.csv(input_file, header=FALSE)

for(i in 1:nrow(origin_data)){   # row_num
  tmp <- as.numeric(origin_data[i,2: given_day_num+1])
  tmp_2 <- ts(tmp, start=1)
  fit <- auto.arima(tmp_2)
  result <- forecast(fit, h=predict_day_num)
  name <- as.vector(origin_data[i, 1])
  tmp_1 <- data.frame(result)$Point.Forecast
  cat(name, tmp_1, file=output_file, append=TRUE, sep=',')
  cat('\n', file=output_file, append=TRUE, sep='')
}
```

在 python 中调整 R 中的预测结果，做精预测。实现代码为：

```
def get_song_predict(in_fl, out_fl, songs):
    result = read_csv(in_fl, header=None, index_col=0)
    result[result  < 0] = 0
    result = pd.merge(result, songs[['song_id', 'artist_id']],
how='left', left_index=True, right_on='song_id')
    result.drop('song_id',
axis=1).groupby('artist_id').sum().to_csv(out_fl, header=None)
    return
```

2. 针对用户进行建模

（1） 计算出每位用户六个月的总播放量、总下载量、总收藏量。

（2） 将用户总播放量完全相同的用户视为同一个用户，将他们的日志合并，相当于将原先的 35 万用户进行合并。针对这些新用户，计算出“新用户-歌手”每天的播放量。

（3） 使用 arima 模型预测，并将预测为负值的设置为 0，并合并相同的歌手。

具体的实现过程与针对歌曲建模的步骤一致，仅在实现的细节上有不同，这里不再重复贴代码。具体是现在 Python 中做数据的预测处理，然后对处理后的数据在 R 语言中建 arima 模型，并用该模型做粗预测，最后再在 Python 中做调整，得到精预测结果。

实验中我们仅保留了播放量最高的前 50 位，下面对结果进行展示。

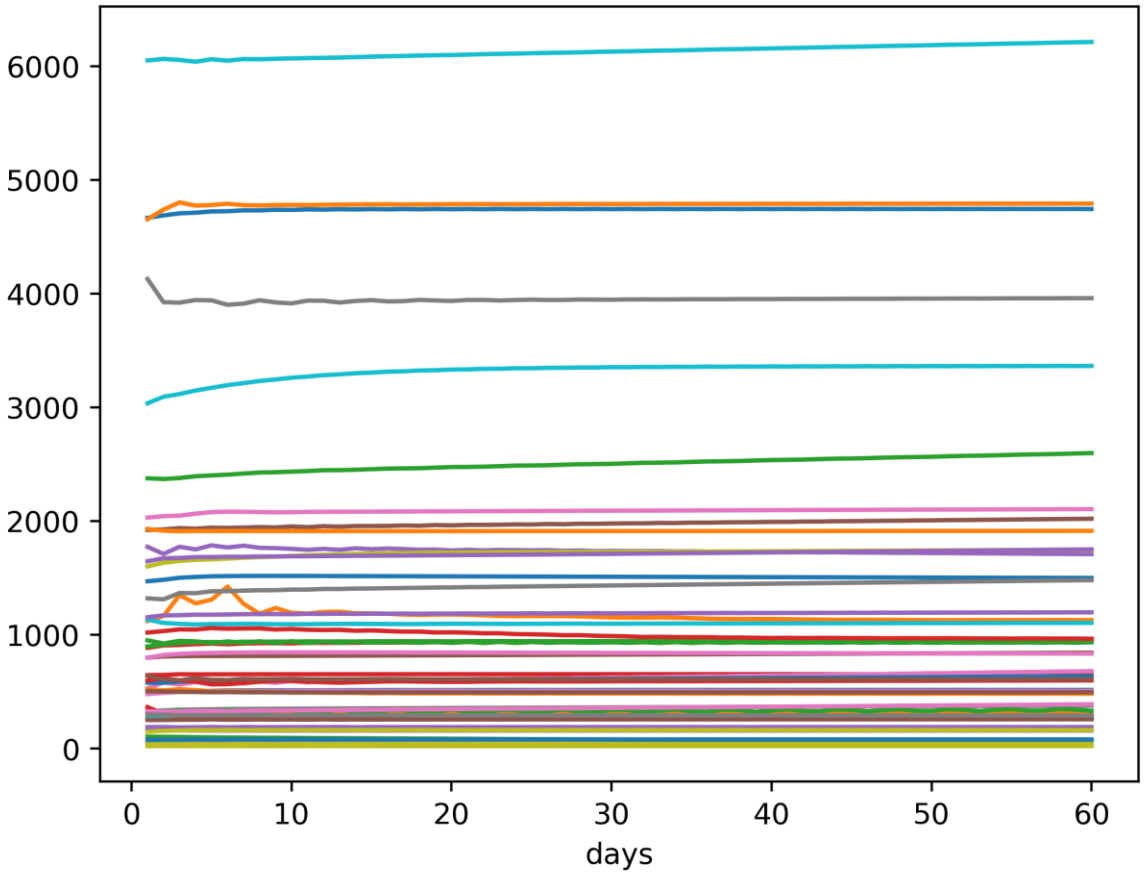
结果展示

后 60 天的歌曲预测结果展示（部分）

	V1	V2	V3	V4	V5	V6	V7	V8
1	023406156015ef87f99521f3b343f71f	874.16444	885.62497	877.01695	872.95485	875.91400	874.04537	875.72990
2	025943df23acf5d3863b35fa6e0d79ec	488.00405	501.75837	504.11437	506.27881	508.18212	511.03177	511.60160
3	03c6699ea836decbc5c8fc2dbae7bd3b	1122.33374	1166.70871	1348.97414	1276.70014	1309.37042	1424.52945	1273.17688
4	099cd99056bf92e5f0e384465890a804	2375.94158	2370.19317	2378.74559	2394.39282	2401.74736	2408.29459	2418.08106
5	0c80008b0a28d356026f4b1097041689	646.05085	647.93960	653.93125	652.90570	653.43792	653.37187	653.67870
6	1339f978614ff19cd48f07e5420556d4	159.84402	160.76258	160.44432	159.60955	158.66870	158.66560	158.92617
7	1731019fbaa825714d5f8e61ad1bb7ff	799.17633	810.62444	812.49183	814.12335	813.80104	814.19282	814.44293
8	25739ad1c56a511fcac86018ac4e49bb	529.97037	581.64481	563.40682	584.32073	566.82024	591.13599	573.25677
9	28e32be6ba67e0c6fdfdff80ce07dfd4	325.31322	335.78288	340.71209	344.01823	346.01587	347.32739	348.41314
10	2b7fedeea967becd9408b896de8ff903	21.17096	23.58749	22.64407	22.38387	22.28152	22.26473	22.25153
11	2e14d32266ee6b4678595f8f50c369ac	6049.38669	6062.97841	6053.86976	6038.00601	6059.80954	6047.61706	6061.33820
12	2ec1450a1389d4e3fc2a9a76c9378bb3	1471.10552	1484.37435	1502.09101	1509.45310	1514.68861	1516.71235	1518.05151
13	33fd0a2cfcfd24e114707bba71ca1de9	301.58153	316.24512	304.68126	314.30651	306.03245	311.78798	307.26163
14	34bc4bc49a725f2686a5cf9c89985798	304.19373	326.23493	341.17699	308.23584	317.64829	342.65685	324.66922
15	3964ee41d4e2ade1957a9135afe1b8dc	886.13743	907.02359	911.15633	918.60318	923.09019	917.67573	924.26996
16	3e395c6b799d3d8cb7cd501b4503b536	1773.91166	1711.58148	1774.40978	1751.31588	1787.07118	1767.81902	1784.13310
17	40bbb0da5570702dd6ff3af5e9e3aea6	1921.84851	1926.93702	1938.73470	1933.70013	1942.29105	1940.33691	1943.34717
18	445a257964b9689f115a69e8cc5dcb75	480.06575	490.76842	499.43925	504.02763	505.58594	506.15665	506.98524
19	4b8eb68442432c242e9242be040bacf9	4128.23059	3924.35111	3920.25243	3943.39446	3940.27376	3901.88479	3911.68016
20	4ee3f9c90101073c99d5440b41f07daa	48.95994	61.08985	53.03143	60.05373	53.31559	59.44168	53.22501
21	4f3f68292e50f39cd8142733512c07d6	1133.76424	1105.16666	1096.45067	1091.65334	1094.34357	1095.07126	1096.69223
22	53dd7de874e0999634c28cdd94d21257	579.07906	578.85676	589.35143	591.78585	593.75675	593.71481	594.63719
23	5e2ef5473cbbdb335f6d51dc57845437	1930.91337	1915.78938	1911.41278	1912.85001	1912.46891	1913.22119	1912.86755
24	61dfd882204789d7d0f70fee2b901cef	105.17392	104.15073	102.93526	100.01972	99.11624	97.31496	96.79055
25	6a493121e53d83f9e119b02942d7c8fe	597.86646	617.20640	599.72722	590.89561	564.95305	566.86076	577.09667
26	6bb4c3bbdb6f5a96d643320c6b6005f5	1154.88068	1171.31247	1172.99358	1177.85770	1177.57001	1179.08922	1182.01087

由可视化的结果可以看出，同一首歌曲，在之后的两个月内每天的播放量波动不大。这也与一般规律相符，一首刚出现的一段时间，受宣传和歌曲本身质量的影响，初始的波动比较大，而当 歌曲出现一段时间后，大众对于歌曲有了相对固定和一致的认知时，歌曲的播放量就趋于稳定了。为了更直观地验证这一分析，我们对上述预测结果做了可视化展示。

其可视化的结果如下：



由上图可以发现，每首歌曲在之后两个月的播放量趋近与直线，这就说明歌曲每天的播放量不大，与上述分析一致。 当然，从此图我们还可以清晰地看出当先最热门的歌曲是那几首，从而由此把握近段时间的音乐流行趋。同时，根据最热门的歌曲，也可以得知当下最潮流和热门的歌手

实验总结

在本项目中，为了得到更好的预测效果，我们进行了多次实验，针对表现出来的问题，不断进行优化。我们发现：

1. 使用歌曲，而不是歌手进行预测，可以得到更好的预测效果。原因可能是歌曲更加细腻，而且数据量更大。
2. 用户合并帮助提高了预测效果。平台 35 万用户里，有 25 万用户的个人总播放行为少于 10 次。理论上，播放量较少的很难个人预测，而将这些用户的日志放在一起，却效果很大，给预测带来了很大的信息增益。

另外，通过本实验，我们进一步了解到数据的挖掘的一般思路：获取数据 -> 分析需求 ->

数据预处理 -> 数据建模 -> 进一步数据分析。通过本次试验，也了解到更多的数据挖掘的方法，现在可以对数据集进行进行简单的规则挖掘，可能会今后自己本方向研究提供思路或有益的帮助。