用 R 语言做逻辑回归预测点击率

理论

逻辑回归

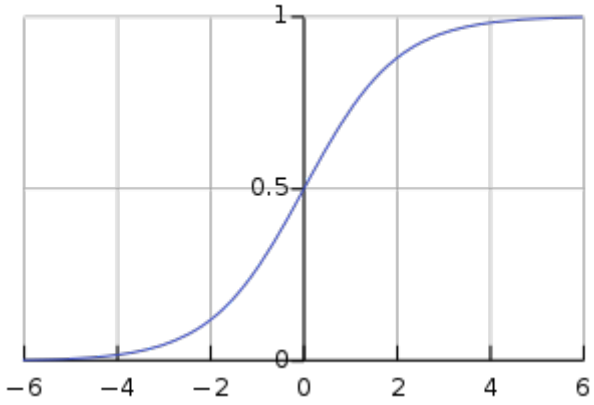
逻辑回归（Logistics Regression，以下简称 LR）是一个分类器，是给定样本特征值 x，以 θ

为参数求解 y=1 概率的过程，为监督学习的一种方法，是一个广义线性模型。其中 x 可以看成多个特征组成的向量，θ 为每个特征的参数，进而转化为多元线性模型

$\theta_0 x_0+\theta_1 x_1+...+\theta_n x_n$ ，于是求逻辑回归的关键变成为一个求线性模型最小化其代价函数的过程。假设 LR 的预测函数为

$h_\theta(x)$ ，其值域为 $0\leqslant h_\theta(x)\leqslant 1$

而 $h_\theta=g(\theta^T x)$ ，这要求 g 函数的值域为 0 到 1，考虑 sigmoid 函数，其函数有以下性质



即 x 越大越趋向 1，越小越趋向 0，当 x=0 时，y=0.5 为决策面，此时为真或假的概率都为 0.5。任何时，真假概率相加都为 1。sigmoid 函数的形式如下：

$$g(z)=\frac{1}{1+e^{-z}}$$

对 sigmoid 函数换元即得逻辑回归的新预测函数：

$$h_\theta(x)=\frac{1}{1+e^{-\theta^T}}x$$

也就是 x 以 θ 为参数的条件下 y 的概率问题。

在求解逻辑回归代价函数的过程中，也就是求拟合函数，有最小二乘法和极大似然法。这里介绍极大似然法也是通常的做法：

我们知道线性回归的代价函数为：

$J(\theta)=\frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}(h_\theta(x^{(i)})-y^{(i)})^2$ ，简化形式为 $Cost(h_\theta(x^{(i)}),y)$

但逻辑回归的预测函数是 $\frac{1}{1+e^{-\theta^T}}x$

代入后最终代价函数是一个非凸函数，这样就没有办法使用梯度下降法进行求解了，使用梯度下降法的好处是在机器学习中采用 map reduce 的思想并行分摊到多个计算节点进行处理。这里就可以根据统计学的最大似然法的思想做变通转为凸函数，把求最大似然转换为求最小化负对数似然。从损失函数角度说，其实就是 log 损失函数，转为对数的好处一是成为负 log 是凸函数，二是把连乘变成加法。下面给出推导，对于单个样本伯努利分布的后验概率有：

$$p(y|x;\theta)=(h_\theta(x))^y(1-h_\theta(x))^{1-y}$$

其中 y=0 或 1

其极大似然函数为：

$$L(\theta|x;y)=\prod_{i=1}^{m}p(y^{(i)}|x^{(i)};\theta) = \prod_{i=1}^{m}(h_\theta(x))^{y^{(i)}}(1-h_\theta(x))^{1-y^{(i)}}$$

log 似然为：

$$l(\theta)=\log(L(\theta|x;y))$$

根据最大化 log 似然等同于最小化-log 似然的方法，可改写代价函数为

$Cost(h_\theta(x),y)=-\log(h_\theta(x))$ 当 y=1

$Cost(h_\theta(x),y)=-\log(1-h_\theta(x))$ 当 y=0

写为更紧凑的格式

$$Cost(h_\theta(x),y)=-ylog(h_\theta(x))-(1-y)*\log(1-h_\theta(x))$$

所以现在的完整的逻辑回归的代价函数是

$$J(\theta)=\frac{1}{m}Cost(h_\theta(x^{(i)},y^{(i)}))=\frac{-1}{m}[\sum_{i=1}^{m}y^{(i)}logh_\theta(x^{(i)})+(1-y^{(i)})*\log(1-h_\theta(x^{(i)}))]$$

随后就可用梯度下降法进行求解。在 R 语言中 glm 做拟合默认采用的是最小二乘法，而 glmnet 包可以更快求解，且使用了梯度下降法，介于安装等环境依赖，这里还是用 glm，对于更高性能的计算，建议采用 glmnet 等高级优化方法。

点击率预测

计算广告的核心之一是 ctr（click through rate）预估，也就是点击率预估。价值在于，广告平台想要赚更多的钱，就要寻找一种机制让竞价更高（当然赚钱也越多）的广告排在最先，有这样一个排名 rank=ctr*bid_price。其中 rank 为排名，ctr 为点击率，点击率等于点击次数/展示次数，而 bid_price 是广告主竞价广告的价格，而作为目前程序化广告交易的热门领域 RTB（实时竞价），作为 dsp（需求方平台）的重要组件，点击率预估系统是否精准成为着为广告主和 dsp 平台获取更多点击转换和收入核心问题，是计算广告领域研究的热门课题。

实战

数据集

数据集采用dsp行业领先企业品友互动在http://contest.ipinyou.com/上放出的专门供ctr预估竞赛的数据集，我们这里由于条件限制，对训练集和测试集没有采取交叉验证，只采用第二季的imp.20130606.txt和clk.20130606.txt,imp.20130607.txt和clk.20130607.txt，为了便于计算，对数据做了预处理，把clk日志（也就是点击日志）中存在的条目的记录在imp日志的最后一列加入isClicked的标记，最后一位为0没有点击，最后一位为1有点击。判断是否存在只要看第一列，也就是clk日志的bidId列的值是否存在于imp日志的bidId列即可。这样我们得到这两个新的文件imp.20130606.txt.new和imp.20130607.txt.new，分别作为训练集和测试集，此外对于原数据的Timestamp由于数据过于离散，做了区间化，根据其时段分别降维成00,01,02,03,04,05,06,...23,而对于UserAgent，我们主要关心操作系统的大版本，所以降维到0,1,2,3，0为android，1为ios，2为macosx，3为其他。同时对用户标签采取特征转换，转换为数字便于计算。关于数据的预处理可以参见convert.php文件，也可以不用关心，因为提供的数据已经是处理好的。

在这里个链接https://pan.baidu.com/s/1mhQyFuK提供数据集下载。

数据格式

"BidID", "Timestamp", "LogType", "iPinYouID", "UserAgent", "IP", "Region", "City", "AdExchange", "Domain", "URL",

"AnonymousURLID", "AdSlotID", "AdSlotWidth", "AdSlotHeight", "AdSlotVisibility", "AdSlotFormat", "AdSlotFloorPrice", "CreativeID",

"BiddingPrice", "PayingPrice", "KeyPageURL", "AdvertiserID", "UserTags", "isClicked"分别代表竞价id，时间，日志类型，cookieId，用户代理，IP地址，省，市，adExchange平台id，域名，url，nimingurl id，广告位id，广告位宽度，广告位高度，广告位可见性，广告位格式，广告位地板价，物料创意id，竞价价格，出价价格，关键页url，广告主id，用户标签，是否点击。

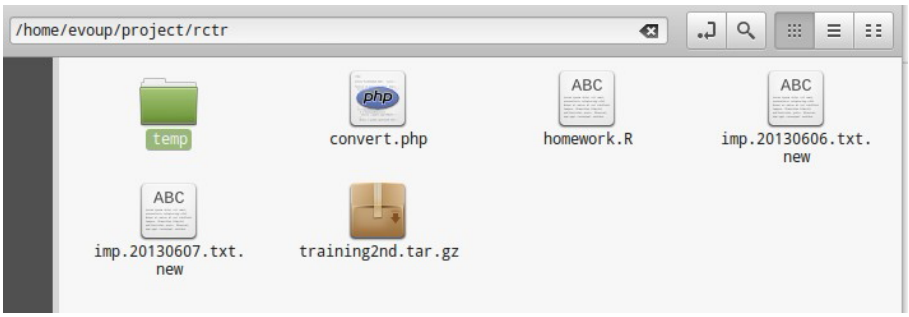
由于时间有限，应该做的数据归一化没有做。

首先把数据读入，由于数据量比较大，使用默认传统read.csv的方法加载竞赛数据集的一个展示日志文件R就不能工作了，我们采用ff这个包，使得即使R不用集成hadoop或spark也可以在本地处理大数据（相对较大，实测还是有限制的）的一个包，方便了环境。

编写的homework.R文件为我们的R源代码，首先需要指定工作目录：

```
Source on Save
1  #predict ctr
2  #dataset download https://pan.baidu.com/s/1mhQyFuK
3
4  mydir = "/home/evoup/project/rctr/"
5  inst_pkgs = load_pkgs =  c("ff","ffbase","biglm")
6  inst_pkgs = inst_pkgs[!(inst_pkgs %in% installed.packages()[,"Package"])]
7  if(length(inst_pkgs)) install.packages(inst_pkgs)
8  pkgs_loaded = lapply(load_pkgs, require, character.only=T)
9  setwd(mydir)
10
11 getOption("fftempdir")
12 options(fftempdir = paste(mydir,"temp",sep=""))
13
14 #click log column name
```

mydir 为数据集存放的路径，建议存放的路径所在的磁盘空间要比较大，因为还需要在其下创建 temp 目录，这是 ff 包读取大数据作为工作文件的一个目录，需要指定好，设置好基本工作目录后，目录结构应该类似如下：



然后便可以加载训练数据

```
training.data.raw <-
read.table.ffdf(file=paste(mydir,"imp.20130606.txt.new",sep=""), VERBOSE=TRUE,
header=FALSE, sep="\t",colClasses=NA, na.strings=c("null"))
```

这是加载测试数据，要加载个几十秒，因为数量较大

```
test.data.raw <-
read.table.ffdf(file=paste(mydir,"imp.20130607.txt.new",sep=""), VERBOSE=TRUE,
header=FALSE, sep="\t",colClasses=NA, na.strings=c("null"))
```

原是日志是没有表头的 csv 文件，要重新赋予列名

```
#click log column name

column_names <-c("BidID", "Timestamp", "LogType", "iPinYouID", "UserAgent",
"IP", "Region", "City", "AdExchange", "Domain", "URL",

"AnonymousURLID", "AdSlotID", "AdSlotWidth", "AdSlotHeight", "AdSlotVisibility",
"AdSlotFormat", "AdSlotFloorPrice", "CreativeID",

"BiddingPrice", "PayingPrice", "KeyPageURL", "AdvertiserID", "UserTags",
"isClicked")

names(training.data.raw)= column_names

names(test.data.raw)= column_names
```

回归因子的选则，对于过于稀疏的数据，通常可以进行降维重新组合特征，但是竞赛中提供的bidId，logType，iPinyouId（cookieId），ip，Domain，URL，AnonymousURLID，AdSlotID，CreativeID，KeyPageURL 是由于过于人工特征提取不容易，这里直接去掉。所以剩下这些特征供计算：

Timestamp，UserAgent，Region，City，AdExchange，AdSlotWidth，AdSlotHeight，AdSlotVisibility，AdSlotFormat，AdSlotFloorPrice，BiddingPrice，PayingPrice，AdvertiserID，AdvertiserID，isClicked

大致看一下数据

	Timestamp	UserAgent	Region	City	AdExchange	AdSlotWidth	AdSlotHeight	AdSlotVisibility
1	0	3	106	117	1	950	90	0
2	0	3	308	320	2	336	280	2
3	0	3	94	95	1	950	90	0
4	0	3	94	95	1	300	250	2
5	0	0	216	217	1	300	250	0
6	0	3	80	85	2	336	280	2
7	0	3	15	23	2	300	250	2
8	0	3	393	393	1	950	90	0
9	0	3	27	28	2	200	200	0
10	0	3	298	299	2	250	250	2
11	0	3	216	219	2	250	250	2
12	0	3	146	152	1	950	90	0
13	0	3	333	334	2	250	250	2

Showing 1 to 13 of 1,821,350 entries

AdSlotFormat	AdSlotFloorPrice	BiddingPrice	PayingPrice	AdvertiserID	UserTags	isClicked
1	0	227	207	3427	1550995684	0
0	5	238	72	3358	133889371	0
1	0	227	108	3358	131915652	0
1	0	300	81	3386	3677931430	0
5	0	227	89	3358	644654711	0
0	5	300	5	3386	362523128	0
0	5	238	126	3358	3566309742	0
1	0	227	11	3427	3566309742	0
0	16	238	50	3427	2692094940	0
0	5	300	64	3386	1769100027	0
0	5	300	217	3386	4224068901	0
1	0	227	86	3427	1674467855	0
0	5	300	64	3386	642336525	0

缺失值处理，暂时没做。

最重要的一步，使用 glm 进行逻辑回归生成模型，采用全部保留特征

```
model <- glm(isClicked ~.,family=binomial(link='logit'),data=data)
```

需要计算一会儿，查看模型

```
summary(model)
```

```
Call:
glm(formula = isClicked ~ ., family = binomial(link = "logit"),
    data = data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.4837  -0.0354  -0.0301  -0.0263   4.1139

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)     -6.036e+00  4.785e-01 -12.613  < 2e-16 ***
Timestamp        9.404e-03  4.657e-03   2.019 0.043472 *
UserAgent       -8.404e-01  3.537e-02 -23.758  < 2e-16 ***
Region           1.785e-03  2.928e-03   0.610 0.542143
City            -1.078e-03  2.899e-03  -0.372 0.710117
AdExchange       2.752e-01  5.314e-02   5.178 2.24e-07 ***
AdSlotWidth     -7.472e-05  1.774e-04  -0.421 0.673646
AdSlotHeight    -5.576e-04  4.077e-04  -1.368 0.171393
AdSlotVisibility -9.614e-04  1.247e-03  -0.771 0.440848
AdSlotFormat     4.509e-01  2.867e-02  15.724  < 2e-16 ***
AdSlotFloorPrice  2.031e-03  9.847e-04   2.063 0.039126 *
BiddingPrice    -1.115e-04  1.202e-03  -0.093 0.926097
PayingPrice      6.332e-03  4.248e-04  14.907  < 2e-16 ***
AdvertiserID    -1.686e-04  4.343e-05  -3.882 0.000103 ***
UserTags         5.573e-11  2.378e-11   2.343 0.019116 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 19407  on 1821349  degrees of freedom
Residual deviance: 18427  on 1821335  degrees of freedom
AIC: 18457
```

可以看到 Timestamp 经过特征提取后还是对模型有共享 0.043472<0.05，Region0.542143 和 City0.710117 贡献不大，广告位特征方面 AdSlotWidth、AdSoltHeight 和 AdSlotVisibility 均贡献不大，也就是说这几个特征对是否点击没有显著的关系，这也很正常毕竟广告位不能决定创意的质量，创意 id 特征太过稀疏这里不好计算没有加入，UserTags 经过特征变化后效果还是不错的。为何 BiddingPrice 效果不尽如人意，理论上来说广告主出价越高应该越容易被点击，这个是要继续考察的。综合来讲，Timestamp,UserAgent,AdExchange,AdSlotFormat,AdSlotFlootPrice,PayingPrice,AdvertiserId 和 UserTags 这些特征还是选的可以的。最后 AIC 数值高达 18457 认为模型还是要继续调整的，一个较小的 AIC 值被认为是一个不错的模型。

那么让我们去掉一些特征再生成一次模型

```
data <-
subset(as.data.frame(training.data.raw),select=c(2,5,9,17,18,20,21,23,24,25))

data_test <-
subset(as.data.frame(test.data.raw),select=c(2,5,9,17,18,20,21,23,24,25))

model_reduce <- glm(isClicked ~.,family=binomial(link='logit'),data=data)
```

再进行方差分析，可能有点慢：

```
anova(model, model_reduce, test="Chisq")
```

```
> anova(model, model_reduce, test="Chisq")
Analysis of Deviance Table

Model 1: isClicked ~ Timestamp + UserAgent + Region + City + AdExchange +
    AdSlotWidth + AdSlotHeight + AdSlotVisibility + AdSlotFormat +
    AdSlotFloorPrice + BiddingPrice + PayingPrice + AdvertiserID +
    UserTags
Model 2: isClicked ~ Timestamp + UserAgent + AdExchange + AdSlotFormat +
    AdSlotFloorPrice + BiddingPrice + PayingPrice + AdvertiserID +
    UserTags
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1   1821335      18427
2   1821340      18437 -5  -10.256  0.06831 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

p=0.06831 不显著，证明 2 个模型差不多，所以结论是可以使用剔除无用变量后的新模型。

进行预测

在测试集中找一条点击的记录进行预测

```
#0.39% clicked=1

newdata = data.frame(Timestamp=0, UserAgent=1, Region=94, City=96,
AdExchange=1,AdSlotWidth=300,AdSlotHeight=250, AdSlotVisibility=0, AdSlotFormat
= 1,

                    AdSlotFloorPrice=0,BiddingPrice =
227,PayingPrice=168,AdvertiserID=3358, UserTags=3566309742, isClicked=1)

predict(model_reduce, newdata, type="response")
```

```
> newdata = data.frame(Timestamp=0, UserAgent=1, Region=94, City=96,  AdExchange=1,AdSlot
+                     AdSlotFloorPrice=0,BiddingPrice = 227,PayingPrice=168,AdvertiserI
> predict(model_reduce, newdata, type="response")
          1
0.003874889
```

点击率为 0.387%，代表大约展示 258 次会有一次点击。

再找一台不惦记的记录进行预测

```
#0.04% clicked=0

newdata = data.frame(Timestamp=0, UserAgent=4, Region=65, City=75,
AdExchange=2,AdSlotWidth=728,AdSlotHeight=90, AdSlotVisibility=2, AdSlotFormat =
0,

                   AdSlotFloorPrice=5,BiddingPrice =
238,PayingPrice=236,AdvertiserID=3427, UserTags=3839198970, isClicked=0)

predict(model_reduce, newdata, type="response")
```

```
> newdata = data.frame(Timestamp=0, UserAgent=4, Region=65, Cit
+                     AdSlotFloorPrice=5,BiddingPrice = 238,Pa
> predict(model_reduce, newdata, type="response")
           1
0.0004201563
```

点击率为 0.042%，代表大约 2381 次展示才有 1 次点击。

至此，点击率 rank 的排序根据以上数值大小已经可以得到评分。可能要问准确率呢？由于逻辑回归的决策边界默认是 0.5 也就是 50%，可见大多的点击都是不准确的，这时其他的决策边界可能是更好的选则，我们调整为第一个能被点击到的概率 0.387%，并用测试集来验证准确率。

```
fitted.results <-
predict(model_reduce,newdata=subset(data_test_clicked),type='response')

fitted.results <- ifelse(fitted.results > 0.000387,1,0)


misClasificError <- mean(fitted.results != data_test_clicked$isClicked)

print(paste('Accuracy',1-misClasificError))
```

```
> fitted.results <- predict(model_reduce,newdata=subset(data_test_clicked),type='response')
> fitted.results <- ifelse(fitted.results > 0.000387,1,0)
> misClasificError <- mean(fitted.results != data_test_clicked$isClicked)
> print(paste('Accuracy',1-misClasificError))
[1] "Accuracy 0.870801033591731"
```

可以看到，经过修改决策边界这时模型在测试集有 87%的准确率，当然原始模型不是很精确所以才手动调整阈值供演示，更好的做法是寻找更多可靠的特征提高准确率，而不是修改决策边界。

参考文献

https://en.wikipedia.org/wiki/Sigmoid_function