1. Step1：（出发）

①、②、③同时出发带上12天食物走6天

消耗总食物：36天

此时剩余食物：①=②=③=6天

Step2：（送出情报）

②、③各将3天食物给①

消耗总食物：36天

此时剩余食物：②=③=3天，①=12天（已能送出情报）

Step3：（保证不死人）

④带上12天食物出发，走3天与②、③汇合，并给②、③各3天食物

消耗总食物：48天

此时剩余食物：②=③=④=3天（三人都能成功回到荒岛）

1. 已知P(X=男)=1/2，P(X=女)=1/2，P(X=色盲|X=男)=5%，P(X=色盲|X=女)=0.25%

所以

1. ①k-means优点：

时间复杂度为线性：，其中t为迭代次数，N为聚类点的个数，适合挖掘大规模数据集；实现简单

②k-means缺点：

对异常值很敏感；对数字特征（身高/体重）取平均值可以，对类别特征（性别/种族）不很适用

③k-medoids优点：

通过构造不相似程度矩阵，解决了对类别特征和异常值的处理难题，更robust

④k-medoids缺点：

时间复杂度提高到，因而只适用于小量数据集

⑤与层次聚类方法的比较优点：

k-means和k-medoids计算量都比层次聚类方法少；k-means和k-medoids中数据点在迭代过程中会跳出原来的初始聚类，而层次聚类不会，这会影响结果准确性

⑥与层次聚类方法的比较缺点：

k-means和k-medoids都需要预先选取k值，而这是一件很难的事，但层次聚类一次就可以得到分为任意k类的结果

1. 1）查询出这批用户的总通话时长：

select A.拨出用户,A.操作时间 as start,B.操作时间 as ending into #temp from phone\_call\_log A,phone\_call\_log B where A.拨出用户=B.拨出用户 and A.操作类型=1 and B.操作类型=2 and A.操作时间<=B.操作时间

declare @x bigint

select @x=SUM(DATEDIFF(second,a.start,a.ending)) from #temp a where not exists(select \* from #temp where a.拨出用户=拨出用户 and a.start=start and a.ending>ending)

select @x as "总通话时长（秒）"

2）查询出这批用户给该运营商带来的收入会减少多少：

declare @y bigint

select @y=SUM(ceiling(DATEDIFF(second,a.start,a.ending)/60.0)) from #temp a where not exists(select \* from #temp where a.拨出用户= 拨出用户 and a.start=start and a.ending>ending)

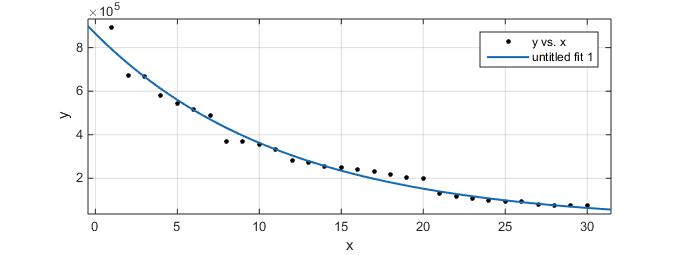
select 1.5\*@y-0.025\*@x as "收入减少（毛）"

1. 从数据图表中可以看出收入随排名衰减很快，不是一般线性关系，故可先假设为指数衰减模型

x=1:30;

y=[890842,672299,668511,578944,541721,516986,488672,369946,367326,355305,330025,282393,272348,252101,248184,238135,231509,216444,203193,197583,129156,117978,108261,96726,92803,92560,79885,77169,76739,76667];

利用MATLAB“应用程序”→“Curve Fitting”工具箱→选择指数拟合y=a\*exp(b\*x)



得到系数a= 8.643e+05, b= -0.08684，R-square: 0.9798说明拟合度很高，假设成立

下面计算App Store中国区在该月开发者总收入total

total=sum(y);

i=31;

while 1

tmp=f(i);

if(tmp<1e-6\*total)

break;

else

total=total+tmp;

end

i=i+1;

end

计算出total约等于 9574228美元，所以当月中国手游市场营收规模约为9574228/0.7/80%≈17096835美元

1. for循环效率不高，没有利用问题的特性，优化如下：

**for (int i=1;i<=n;i+=2)**

**temp-=1;**

**if(n%2!=0)**

**temp+=n+1;**

？？？导致问题的计算机原理？？？

1. 方法：

利用Taylor展开：

得到：，对→收敛

最后再利用换底公式：

Java实现：

public static double log(double a,double b)

{

if(a<=1 || b<=0)

throw new RuntimeException("Invalid argument");

double x\_a=(a-1)/(a+1),x\_b=(b-1)/(b+1);

double ln\_a=2\*x\_a,ln\_b=2\*x\_b,ln\_a\_pre,ln\_b\_pre;

int i=3;

while(true)

{

ln\_a\_pre=ln\_a;

ln\_a+=2.0/i\*Math.pow(x\_a,i);

ln\_b\_pre=ln\_b;

ln\_b+=2.0/i\*Math.pow(x\_b,i);

i+=2;

if(Math.abs(ln\_b/ln\_a-ln\_b\_pre/ln\_a\_pre)<1e-7)

break;

}

return ln\_b/ln\_a;

}

1. 利用数组记录0岁~9岁袋鼠个数，f更新一年后袋鼠个数，N年表示作用N次f

Java实现：

private static double[] f(double[] a)

{

int newborn=0;

for(int i=4;i<a.length-1;i++)

newborn+=a[i];

for(int i=a.length-1;i>0;i--)

a[i]=a[i-1];

a[0]=newborn;

return a;

}

public static double total(double[] start,int N)

{

if(start.length!=10)

throw new RuntimeException("Invalid input");

for(int i=0;i<N;i++)

start=f(start);

int ans=0;

for(int i=0;i<start.length;i++)

ans+=start[i];

return ans;

}

public static void main(String[] args)

{

System.out.println(total(new double[]{0,0,0,0,0,1,0,0,0,0},2000));

}

输出结果：2.147483647E9

1. 手游玩龄百分比加和超出100%应为数据有误，故不设图显示

收入、职业加和不足100%，故再各分别加一档“>4999”和“其他”

1. 首先计算单位广告成本获得的收益：

|  |  |  |  |
| --- | --- | --- | --- |
|  | a渠道 | b渠道 | c渠道 |
| 新增量/万元广告 | 1,471.77 | 2,578.20 | 972.00 |
| 累计消费/万元广告 | 20407.58565 | 23429.545 | 36661.13333 |

我们看到c渠道能获得的收益率最高，事实上观察30日ARPU值，我们可以发现这可能是因为c渠道获得的用户质量比较高，更愿意消费

其次比较各渠道次日留存率：

我们看到a渠道次日留存率很不稳定，b渠道次日留存率比较稳定而且平均水平高于a渠道，c渠道次日留存率稳定度一般但平均水平最高。这可能是因为a渠道推广方式只追求推广数量不追求推广质量故次日留存率低且不稳定，b渠道有针对性的推广那些倾向于玩类似类型游戏的人群故留存率相当稳定，c渠道舍弃一些针对性但又扩大推广人群为所有玩游戏的人群故稳定性稍差一些

最后观察3种类型推广渠道市场状况：

|  |  |  |
| --- | --- | --- |
| 渠道类型 | 现有渠道数量 | Top5市场份额总和(%) |
| A类渠道 | 300+ | 75% |
| B类渠道 | 1000+ | 42% |
| C类渠道 | 86 | 80% |

我们发现A,C类渠道Top5的市场份额总和很高，B类渠道相对较低，故我们可以采取：

A类投Top5五个渠道达到短时间内快速增加用户总量的效果；

C类投Top5五个渠道达到吸引高消费人群的效果；

B类适当多投几十个渠道达到获取稳定用户数，稳定收益率的效果