机器学习导论
习题六

181220031, 李惟康, liwk@smail.nju.edu.cn

2020 年 6 月 11 日

学术诚信

本课程非常重视学术诚信规范，助教老师和助教同学将不遗余力地维护作业中的学术诚信规范的建立。希望所有选课学生能够对此予以重视。[1]

(1) 允许同学之间的相互讨论，但是**署你名字的工作必须由你完成**，不允许直接照搬任何已有的材料，必须独立完成作业的书写过程;

(2) 在完成作业过程中，对他人工作（出版物、互联网资料）中文本的直接照搬（包括原文的直接复制粘贴及语句的简单修改等）都将视为剽窃，剽窃者成绩将被取消。**对于完成作业中有关键作用的公开资料，应予以明显引用**；

(3) 如果发现作业之间高度相似将被判定为互相抄袭行为，**抄袭和被抄袭双方的成绩都将被取消**。因此请主动防止自己的作业被他人抄袭。

作业提交注意事项

(1) 请在 LaTeX 模板中**第一页填写个人的姓名、学号、邮箱信息**；

(2) 本次作业需提交该 pdf 文件、问题 3 可直接运行的源码 (BoostMain.py, RandomForestMain.py，不需要提交数据集)，将以上三个文件压缩成 zip 文件后上传。zip 文件格式为**学号.zip**，例如 170000001.zip；pdf 文件格式为**学号 _ 姓名.pdf**，例如 170000001_ 张三.pdf。

(3) 未按照要求提交作业，或提交作业格式不正确，将会**被扣除部分作业分数**；

(4) 本次作业提交截止时间为 **6 月 11 日 23:59:59**。**本次作业不允许缓交，截止时间后不接收作业，本次作业记零分**。

[1] 参考尹一通老师高级算法课程中对学术诚信的说明。

1 [25pts] Bayesian Network

贝叶斯网 (Bayesian Network) 是一种经典的概率图模型，请学习书本 7.5 节内容回答下面的问题：

(1) **[5pts]** 请画出下面的联合概率分布的分解式对应的贝叶斯网结构：

$$\Pr(A,B,C,D,E,F,G) = \Pr(A)\Pr(B)\Pr(C)\Pr(D|A)\Pr(E|A)\Pr(F|B,D)\Pr(G|D,E)$$

(2) **[5pts]** 请写出图1中贝叶斯网结构的联合概率分布的分解表达式。

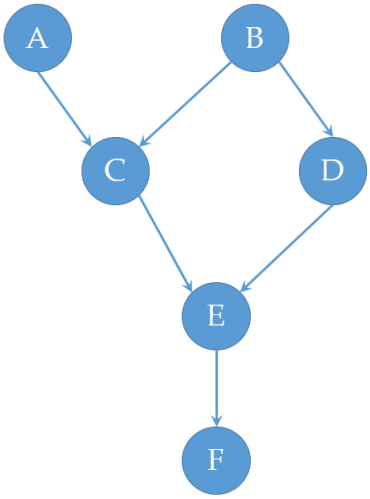

图 1: 题目 1-(2) 有向图

(3) **[15pts]** 基于第 (2) 问中的图1, 请判断表格1中的论断是否正确。首先需要作出对应的道德图，并将下面的表格填完整。

表 1: 判断表格中的论断是否正确

序号	关系	True/False	序号	关系	True/False
1	$A \perp\!\!\!\perp B$	Ture	7	$F \perp B \mid C$	False
2	$A \perp B \mid C$	False	8	$F \perp B \mid C, D$	Ture
3	$C \perp\!\!\!\perp D$	False	9	$F \perp B \mid E$	Ture
4	$C \perp D \mid E$	False	10	$A \perp\!\!\!\perp F$	False
5	$C \perp D \mid B, F$	False	11	$A \perp F \mid C$	False
6	$F \perp\!\!\!\perp B$	False	12	$A \perp F \mid D$	False

Solution.

(1) 贝叶斯网结构如下图2所示.

(2) 图1中贝叶斯网结构的联合概率分布的分解表达式:

$$\Pr(A,B,C,D,E,F) = \Pr(A)\Pr(B)\Pr(C|A,B)\Pr(D|B)\Pr(E|C,D)\Pr(F|E)$$

(3) 道德图如下图3所示，补充完整的表格见表1.

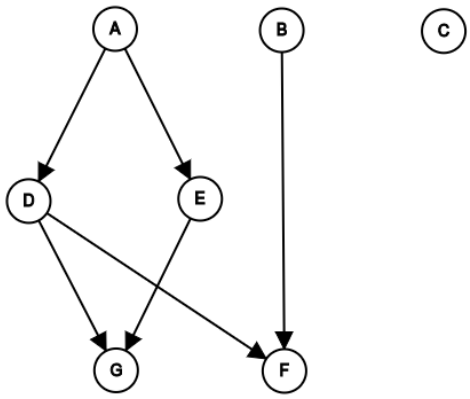

图 2: 贝叶斯网结构

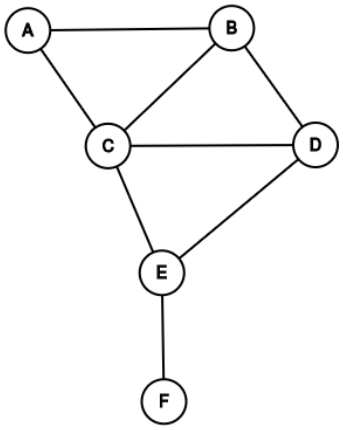

图 3: 道德图

2 [35+10pts] Theoretical Analysis of k-means Algorithm

给定样本集 $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, k-means 聚类算法希望获得簇划分 $\mathcal{C} = \{C_1, C_2, \cdots, C_k\}$ 使得最小化欧式距离

$$J(\gamma, \mu_1, \ldots, \mu_k) = \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} \|\mathbf{x}_i - \mu_j\|^2 \tag{1}$$

其中 $\mu_1, \ldots, \mu_k$ 为 k 个簇的中心 (means), $\gamma \in \mathbb{R}^{n \times k}$ 为指示矩阵 (indicator matrix) 定义如下：若 $\mathbf{x}_i$ 属于第 j 个簇, 则 $\gamma_{ij} = 1$, 否则为 0，则最经典的 k-means 聚类算法流程如算法1中所示

Algorithm 1 $k-$means Algorithm

1: Initialize $\mu_1, \ldots, \mu_k$;

2: **repeat**

3: **Step 1:** Decide the class memberships of $\{\mathbf{x}_i\}_{i=1}^n$ by assigning each of them to its nearest cluster center.

$$\gamma_{ij} = \begin{cases} 1, & \|\mathbf{x}_i - \mu_j\|^2 \leq \|\mathbf{x}_i - \mu_{j'}\|^2, \forall j' \\ 0, & \text{otherwise} \end{cases}$$

4: **Step 2:** For each $j \in \{1, \cdots, k\}$, recompute μ_j using the updated γ to be the center of mass of all points in C_j :

$$\mu_j = \frac{\sum_{i=1}^{n} \gamma_{ij} \mathbf{x}_i}{\sum_{i=1}^{n} \gamma_{ij}}$$

5: **until** the objective function J no longer changes;

(1) **[5pts]** 试证明, 在算法1中, Step 1 和 Step 2 都会使目标函数 J 的值降低.

(2) **[5pts]** 试证明, 算法1会在有限步内停止。

(3) **[10pts]** 试证明, 目标函数 J 的最小值是关于 k 的非增函数, 其中 k 是聚类簇的数目。

(4) **[15pts]** 记 $\hat{\mathbf{x}}$ 为 n 个样本的中心点, 定义如下变量,

$$\begin{aligned} T(X) &= \sum_{i=1}^{n} \|\mathbf{x}_i - \hat{\mathbf{x}}\|^2 / n \\ W_j(X) &= \sum_{i=1}^{n} \gamma_{ij} \|\mathbf{x}_i - \mu_j\|^2 / \sum_{i=1}^{n} \gamma_{ij} \\ B(X) &= \sum_{j=1}^{k} \frac{\sum_{i=1}^{n} \gamma_{ij}}{n} \|\mu_j - \hat{\mathbf{x}}\|^2 \end{aligned}$$

试探究以上三个变量之间有什么样的等式关系？基于此请证明, k-means 聚类算法可以认为是在最小化 $W_j(X)$ 的加权平均, 同时最大化 $B(X)$.

(5) **[Bonus 10pts]** 在公式1中, 我们使用 ℓ_2- 范数来度量距离 (即欧式距离), 下面我们考虑使用 ℓ_1- 范数来度量距离

$$J'(\gamma, \mu_1, \ldots, \mu_k) = \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} \|\mathbf{x}_i - \mu_j\|_1 \tag{2}$$

- 请仿效算法1, 给出新的算法 (命名为 $k-$means-ℓ_1 算法) 以优化公式2中的目标函数 J'.
- 当样本集中存在少量异常点(outliers)时, 上述的 $k-$means-ℓ_2 和 $k-$means-ℓ_1 算法，我们应该采用哪种算法？即哪个算法具有更好的鲁棒性？请说明理由。

Solution.

(1) 算法1中的 Step 1 根据最近的簇均值向量确定 $\{\mathbf{x}_i\}_{i=1}^n$ 的簇标记，即：

$$\gamma_{ij} = 1, \quad \text{if } \|\mathbf{x}_i - \mu_j\|^2 \leq \|\mathbf{x}_i - \mu_{j'}\|^2, \ \forall j'$$

故显然 Step 1 会使目标函数 J 的值降低；

而 Step 2 将所有簇标记相同的样本点的均值作为新的均值向量，令每一步更新后的第 j 个簇均值向量为：

$$\mu'_j = \frac{\sum_{i=1}^{n} \gamma_{ij} \mathbf{x}_i}{\sum_{i=1}^{n} \gamma_{ij}}, \ \forall j$$

那么有：

$$\begin{aligned} J(\gamma, \mu_1, \ldots, \mu_k) &= \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} \|\mathbf{x}_i - \mu_j\|^2 \\ &= \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} \left\|(\mathbf{x}_i - \mu'_j) + (\mu'_j - \mu_j)\right\|^2 \\ &= \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} \left(\|\mathbf{x}_i - \mu'_j\|^2 + \|\mu'_j - \mu_j\|^2 + 2(\mathbf{x}_i - \mu'_j) \cdot (\mu'_j - \mu_j)\right) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} \|\mathbf{x}_i - \mu'_j\|^2 + n \sum_{j=1}^{k} \gamma_{ij} \|\mu'_j - \mu_j\|^2 \\ &\geqslant \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} \|\mathbf{x}_i - \mu'_j\|^2 = J(\gamma, \mu'_1, \ldots, \mu'_k). \end{aligned}$$

即 Step 2 同样会使目标函数 J 的值降低。

(2) 由 (1) 中可知，算法1的每次迭代都会使得 J 降低，即 J 单调递减的。因此迭代过程中不会出现相同的 γ。由于 γ 具有 nk 个元素，每个元素的值为 0 或 1，因此它具有有限数量的可能值。故算法1会在有限步内停止。

(3) 假设对于 $k=\alpha$，目标函数 J 的最小值关于 k 非增，新引入一个簇中心 $\mu_{\alpha+1}$，易见此时的目标函数仍未收敛到最小值，即此时 J 还需减小以收敛到最小值。因此目标函数 J 的最小值关于 k 非增。

(4) 由题有：

$$\begin{aligned}\sum_{j=1}^{k}\sum_{i=1}^{n}\gamma_{ij}W_j(X)+nB(X) &= \sum_{j=1}^{k}\sum_{i=1}^{n}\gamma_{ij}\left(\|\mathbf{x}_i-\mu_j\|^2+\|\mu_j-\hat{\mathbf{x}}\|^2\right)\\ &= \sum_{j=1}^{k}\sum_{i=1}^{n}\gamma_{ij}\left(\mathbf{x}_i^2+\hat{\mathbf{x}}^2-2\mathbf{x}_i\hat{\mathbf{x}}+2\mathbf{x}_i\hat{\mathbf{x}}-2\mathbf{x}_i\mu_j-2\mu_j\hat{\mathbf{x}}+2\mu_j^2\right)\\ &= \sum_{j=1}^{k}\left(\sum_{i=1}^{n}\gamma_{ij}\|\mathbf{x}_i-\hat{\mathbf{x}}\|^2\right)+k\end{aligned}$$

其中 k 为上式中对应的余项；

$$\begin{aligned}\sum_{j=1}^{k}\left(\sum_{i=1}^{n}\gamma_{ij}\|\mathbf{x}_i-\hat{\mathbf{x}}\|^2\right)+k &= \sum_{j=1}^{k}\left(\sum_{i=1}^{n}\gamma_{ij}\|\mathbf{x}_i-\hat{\mathbf{x}}\|^2\right)+k\\ &= n\sum_{i=1}^{n}\|\mathbf{x}_i-\hat{\mathbf{x}}\|^2+k\\ &= n^2T(X)+k\end{aligned}$$

综上可得：

$$\sum_{j=1}^{k}\sum_{i=1}^{n}\gamma_{ij}W_j(X)+nB(X)=n^2T(X)+k$$

由于：$\sum_{j=1}^{k}\sum_{i=1}^{n}\gamma_{ij}W_j(X)=J\left(\gamma,\mu_1,\ldots,\mu_k\right)$，即在 $k-$means 迭代过程中最小化，而 $n^2T(X)$ 为常数，则 $B(X)$ （近似）最大化。

(5) ℓ_1- 范数度量下的 $k-$means 算法2伪代码如下：

其中，ℓ_1 距离度量下，为了最小化 (2) 式：

$$J'\left(\gamma,\mu_1,\ldots,\mu_k\right)=\sum_{i=1}^{n}\sum_{j=1}^{k}\gamma_{ij}\|\mathbf{x}_i-\mu_j\|_1$$

考虑仅对 μ_j 的目标函数：$J^*=\sum_{i=1}^{n}\gamma_{ij}\|\mathbf{x}_i-\mu_j\|_1$ 求导得：

$$\frac{\partial J^*}{\partial\mu_j}=\sum_{i=1}^{n}\gamma_{ij}\mathrm{sgn}\left(\mathbf{x}_i-\mu_j\right)$$

注意到当 $\mathbf{x}_i>\mu_j$ 时 $\mathrm{sgn}\left(\mathbf{x}_i-\mu_j\right)=1$；当 $\mathbf{x}_i<\mu_j$ 时 $\mathrm{sgn}\left(\mathbf{x}_i-\mu_j\right)=-1$。只有当 $\mathrm{sgn}\left(\mathbf{x}_i-\mu_j\right)$ 一式当中 1 和-1 的数目相等时上式中的偏导数为 0，即使得目标函数 J^* 最小化，因此 μ_j 应取中位数：

$$\mu_j=\mathrm{median}(x_j|\gamma_{ij}=1)$$

Algorithm 2 $k-$means-ℓ_1 Algorithm

1: Initialize $\mu_1, \ldots, \mu_k$;

2: **repeat**

3: **Step 1:** Decide the class memberships of $\{\mathbf{x}_i\}_{i=1}^n$ by assigning each of them to its nearest cluster center.

$$\gamma_{ij} = \begin{cases} 1, & \|\mathbf{x}_i - \mu_j\|_1 \leq \|\mathbf{x}_i - \mu_{j'}\|_1, \forall j' \\ 0, & \text{otherwise} \end{cases}$$

4: **Step 2:** For each $j \in \{1, \cdots, k\}$, recompute μ_j using the updated γ to be the center of mass of all points in C_j :

$$\mu_j = \text{median}(x_j | \gamma_{ij} = 1)$$

5: **until** the objective function J no longer changes;

存在少量异常点时应当考虑使用 $k-$means-ℓ_1 算法，显然异常点的存在对于平均值的影响要大于对于中位数的影响，即 $k-$means-ℓ_1 算法具有更好的鲁棒性。

3 [40pts] Coding: Ensemble Methods

本次实验中我们将结合两种经典的集成学习思想：Boosting 和 Bagging，对集成学习方法进行实践。本次实验选取 UCI 数据集 Adult，此数据集为一个二分类数据集，具体信息可参照链接，为了方便大家使用数据集，已经提前对数据集稍作处理，并划分为训练集和测试集，数据集文件夹为 adult_dataset。

由于 Adult 是一个类别不平衡数据集，本次实验选用 AUC 作为评价分类器性能的评价指标，可调用sklearn 算法包对 AUC 指标进行计算。

(1) 本次实验要求使用 Python3 编写，要求代码分布于两个文件中，BoostMain.py, RandomForestMain.py ，调用这两个文件就能完成一次所实现分类器的训练和测试；

(2) **[35pts]** 本次实验要求编程实现如下功能：

- **[10pts]** 结合教材 8.2 节中图 8.3 所示的算法伪代码实现 AdaBoost 算法，基分类器选用决策树，基分类器可调用 sklearn 中决策树的实现；
- **[10pts]** 结合教材 8.3.2 节所述，实现随机森林算法，基分类器仍可调用 sklearn 中决策树的实现，也可以手动实现，在实验报告中请给出随机森林的算法伪代码；
- **[10pts]** 结合 AdaBoost 和随机森林的实现，调查基学习器数量对分类器训练效果的影响，具体操作如下：分别对 AdaBoost 和随机森林，给定基分类器数目，在训练数据集上用 5 折交叉验证得到验证 AUC 评价。在实验报告中用折线图的形式报告实验结果，折线图横轴为基分类器数目，纵轴为 AUC 指标，图中有两条线分别对应 AdaBoost 和随机森林，基分类器数目选取范围请自行决定；

- **[5pts]** 根据参数调查结果，对 AdaBoost 和随机森林选取最好的基分类器数目，在训练数据集上进行训练，在实验报告中报告在测试集上的 AUC 指标；

(3) **[5pts]** 在实验报告中，除了报告上述要求报告的内容外还需要展现实验过程，实验报告需要有层次和条理性，能让读者仅通过实验报告便能了解实验的目的，过程和结果。

实验报告.

1. 实验目的

本次实验主要实现了两种集成学习算法：Adaboost 与 Random Forest 算法，实现后首先在测试集上进行性能测试，通过调参等手段对性能进行优化。之后通过交叉验证探究基学习器 (这里为决策树) 数目与性能之间的关系以寻求最优基分类器数目，以此加深对于上述两种算法的理解与应用。

2. 实验过程

i. AdaBoost: 根据教材上的相应伪代码不难实现这一算法，主要思路是定义 `AdaBoost` 类，其中有 `fit`，`predict`，`predict_proba`，`score` 四个成员函数，分别代表训练、预测数据集对应的标签，预测数据集中样本属于**正例**的概率以及分类的准确性。(基分类器采用 `sklearn.tree.DecisionTreeClassifier`) 需要特别注意的是，训练过程中如果基分类器的误差为 0，则应当立即将其加入基分类器的列表中，将其对应权重置为 1，并立即停止训练；另外值得一提的是在预测样本属于正例的概率时，为避免 n 个基分类器预测结果分别属于正例的概率之和大于 1，应当对概率作加权平均的处理，即:

```
prob += (self.alpha[t]/self.alpha.sum())*self.base[t].predict_proba(X)[:,1]
```

ii. RandomForest: 这一算法的实现思路与 AdaBoost 类似,主要思路同样是定义 `RandomForest` 类，其中有 `bootstrap_sample`，`fit`，`predict`，`predict_proba`，`score` 五个成员函数，分别代表自助采样、训练、预测数据集对应的标签，预测数据集中样本属于**正例**的概率以及分类的准确性。(基分类器采用 `sklearn.tree.DecisionTreeClassifier`) 其中 `fit` 中，根据随机森林算法的要求，对基决策树每个结点应当选择 $k = \log_2 d$ 个属性，从中再选取最优属性进行划分，因此基决策树应当设置为:`DecisionTreeClassifier(max_features="log2")`. 此外，与 AdaBoost 不同，RandomForest 的 `predict` 对多个基分类器的预测结果结合时，直接使用投票法即可：

```
def predict(self, X):
        m = X.shape[0]
        votes = np.zeros((m, self.n_estimators)).astype(int)
        for j in range(self.n_estimators):
            votes[:,j] = self.base[j].predict(X)
        y_pred = np.zeros(m).astype(int)
        for i in range(m):
            (values,counts) = np.unique(votes[i,:],return_counts=True)
            y_pred[i] = values[counts.argmax()]
        return y_pred
```

即用 votes 的每一列存储一个基分类器的预测结果，最后按行 (即对每一个样本) 进行投票即可。与之相对应的，predict_proba 函数也采取与 predict 类似的思路，只在最终产生每个样本属于正例的概率时对每个基学习器的预测结果进行平均即可：

```
def predict_proba(self, X):
    m = X.shape[0]
    votes = np.zeros((m, self.n_estimators))
    for j in range(len(self.base)):
        votes[:,j] = self.base[j].predict_proba(X)[:,1]
    prob = np.zeros(m)
    for i in range(m):
        prob[i] = votes[i, :].sum()/self.n_estimators
    return prob
```

随机森林算法伪代码如下所示：

Algorithm 3 Random Forest

输入: 训练集 $D = \{(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), \ldots, (\mathbf{x_n}, y_n)\}$;
基学习算法 $\mathfrak{L}$(决策树引入随机属性选择);
训练轮数 T.

输出: $H(X) = \arg\max_{y \in \mathcal{Y}} \sum_{t=1}^{T} \mathbb{I}(h_t(x) = y)$

1: **for** $t = 1, 2, \ldots, T$ **do**
2: $h_t = \mathfrak{L}(D, \mathcal{D}_{bs})$
3: **end for**

3. 实验结果

AdaBoost 与 Random Forest 算法在测试集上的预测结果 (指标为准确率，基学习器数目为 50) 如下所示：

```
(Adaboost) accuracy on testing set:  0.8132178613107303
(RandomForest) accuracy on testing set:  0.8554142865917327
```

经过对于基学习器的参数调整，性能有少许提升：

```
(Adaboost) accuracy on testing set:  0.8559056569006818
(RandomForest) accuracy on testing set:  0.8638290031324857
```

对 AdaBoost 与 RandomForest 使用 (5 折) 交叉验证所得到的基分类器数目与 AUC 指标之间的关系如下图所示：

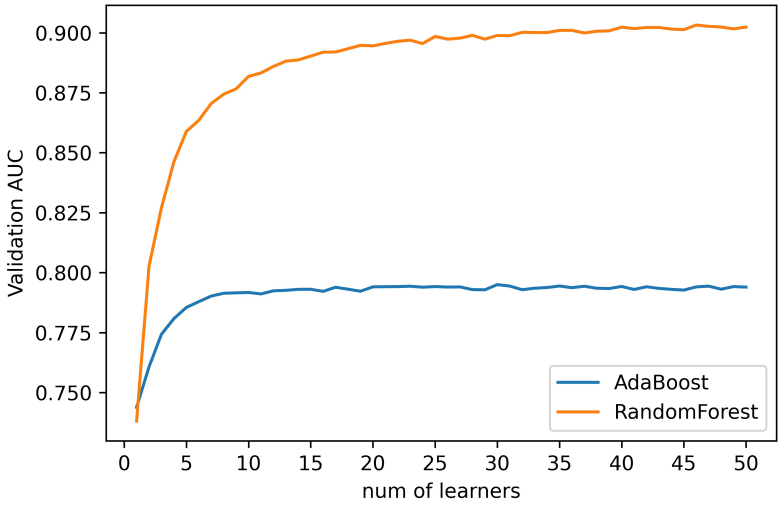

图 4: AdaBoost 与 RandomForest 在不同基分类器数目下的 AUC 指标

由上图 4 所示，对于 AdaBoost 算法，基分类器数目在 1 到 10 之间时，AUC 指标上升较快，超过 10 之后上升非常缓慢，以致在小范围内波动；而对于 RandomForest 算法，基分类器数目在 1 到 15 之间时 AUC 都保持了较快的上升趋势，而在超过 35 之后变得十分缓慢。
根据上述分析，我们选取交叉验证时使得 AUC 指标最好的分类器数目作为最优分类器数目。故对于 AdaBoost 取 30，对于 RandomForest 取 46，可得：

```
NUM=30, AUC= 0.829445127141636
```

```
NUM=46, AUC= 0.9004814008402717
```