Report of Intelligent Data Analysis

E-mail: 11849060@mail.sustech.edu.cn
Tel: +86-156-29096543

高度不平衡数据集分类的 AUC 最大化*

谭树杰 [1]

[1](南方科技大学 计算机科学与技术系,广东 深圳 518055)

摘 要: ROC 面积（AUC）是二分类问题中的一个重要性能指标，它能够衡量模型的分类能力有多好。 对于高度不平衡的数据集，直接最大化 AUC 而不是分类精度具有理论和实际意义。 在这个任务中，我实现了[1]中提出的最大化 AUC 的随机在线算法。

关键词: 分类;AUC 最大化;机器学习；不平衡分类

中图法分类号：TP311

中文引用格式: 谭树杰. 高度不平衡数据集分类的 AUC 最大化.

英文引用格式: Shujie Tan. AUC Maximization of Highly-imbalanced Classification

AUC Maximization of Highly-imbalanced Classification

Shujie Tan [1]

[1](Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China)

Abstract: Area Under ROC(AUC) is an important performance metric in binary classification problem. AUC can tell how good the classification capacity of a model. For highly imbalanced dataset, it is of theoretical and practical interest to maximize the AUC directly rather than the classification accuracy. In this assignment, I implemented a stochastic online algorithm proposed in [1] for AUC maximization.

Key words: Classification;AUC maximization; Machine learning;Imbalanced dataset

First, let's explore the dataset, we can observe that training dataset only include two values, 0 and 1.

```
In [31]:  1  np.unique(X)
Out[31]: array([0, 1], dtype=uint8)
```

Fig. 1

In the following context, we first analyze and perform data preprocessing. Then, we introduce the algorithm. Finally, we conclude briefly

* 课程项目：南方科技大学计算机科学与工程系
Course Project: Department of Computer Science and Engineering, Southern University of Science and Technology

1 Data Preprocessing

The dataset is pretty integrital, therefore we don't need consider missing values.

We use sklearn.preprocessing.MinMaxScaler to transform features of the dataset by scaling each feature to a range (-1,1).

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler((-1,1))
scaler.fit(X)
X = scaler.transform(X)
np.unique(X)
```

```
array([-1.,  1.])
```

The transformation is given by:

```
X_std = (X - X.min(axis=0)) / (X.max(axis=0) - X.min(axis=0))
X_scaled = X_std * (max - min) + min
```

Where min = -1, max = 1

2 The details of implementing the algorithm.

Let the input space $\mathcal{X} \subseteq \mathbb{R}^d$ and the output space $\mathcal{Y} = \{-1, +1\}$ The data $\mathbf{z} = \{(x_i, y_i), i = 1, \ldots, n\}$ are i.i.d. samples drawn from an unknown distribution. The AUC for any scoring function f : X $\rightarrow$ R is equivalent to

$$\mathrm{AUC}(f) = \Pr\left(f(x) \geq f\left(x'\right) | y = +1, y' = -1\right)$$

Where (x, y) and (x', y') are independent drawn from the distribution.

The target is find the optimal decision function f:

$$\begin{aligned} \arg\max_f \mathrm{AUC}(f) &= \arg\min_f \Pr\left(f(x) < f\left(x'\right) | y = 1, y' = -1\right) \\ &= \arg\min_f \mathbb{E}\left[\mathbb{I}\left[\left[f\left(x'\right) - f(x) > 0\right]\right] y = 1, y' = -1\right] \end{aligned}$$

Where $\mathbb{I}(\cdot)$ is the indicator function that takes value 1 if the argument is true and 0 other wise. Since I(·) is not continuous, it is by its convex surrogates loss ℓ_2 loss $(1 - (f(x) - f(x')))^2$

It can be proven it has an equivalent representation as a (Stochastic) Saddle Point Problem (SPP)

$$\min_{u \in \Omega_1} \max_{\alpha \in \Omega_2} \{f(u, \alpha) := \mathbb{E}[F(u, \alpha, \xi)]\}$$

Where we define:

$$\begin{aligned} F(\mathbf{w}, a, b, \alpha; z) = &(1-p)\left(\mathbf{w}^\top x - a\right)^2 \mathbb{I}_{[y=1]} + p\left(\mathbf{w}^\top x - b\right)^2 \mathbb{I}_{[y=-1]} \\ &+ 2(1+\alpha)\left(p\mathbf{w}^\top x \mathbb{I}_{[y=-1]} - (1-p)\mathbf{w}^\top x \mathbb{I}_{[y=1]}\right) - p(1-p)\alpha^2 \end{aligned}$$

The AUC optimization is equivalent to:

$$\min_{\substack{\|\mathbf{w}\| \leq R \\ (a,b) \in \mathbb{R}^2}} \max_{\alpha \in \mathbb{R}} \left\{f(\mathbf{w}, a, b, \alpha) := \int_{\mathcal{Z}} F(\mathbf{w}, a, b, \alpha; z) d\rho(z)\right\}$$

We can solve the problem by calculating the gradient to achieve the saddle.

$$\hat{G}_t(v, \alpha, z) = \left(\partial_v \hat{F}_t(v, \alpha, z), -\partial_\alpha \hat{F}_t(v, \alpha, z)\right)$$

The psudocode is shown below:

Stochastic Online AUC Maximization (SOLAM)
1. Choose step sizes $\{\gamma_t > 0 : t \in \mathbb{N}\}$
2. Initialize $t = 1$, $v_1 \in \Omega_1$, $\alpha_1 \in \Omega_2$ and let $\hat{p}_0 = 0$, $\bar{v}_0 = 0$, $\bar{\alpha}_0 = 0$ and $\bar{\gamma}_0 = 0$.
3. Receive a sample $z_t = (x_t, y_t)$ and compute $\hat{p}_t = \frac{(t-1)\hat{p}_{t-1} + \mathbb{I}_{[y_t=1]}}{t}$
4. Update $v_{t+1} = P_{\Omega_1}(v_t - \gamma_t \partial_v \hat{F}_t(v_t, \alpha_t, z_t))$
5. Update $\alpha_{t+1} = P_{\Omega_2}(\alpha_t + \gamma_t \partial_\alpha \hat{F}_t(v_t, \alpha_t, z_t))$
6. Update $\bar{\gamma}_t = \bar{\gamma}_{t-1} + \gamma_t$
7. Update $\bar{v}_t = \frac{1}{\bar{\gamma}_t}(\bar{\gamma}_{t-1}\bar{v}_{t-1} + \gamma_t v_t)$, and $\bar{\alpha}_t = \frac{1}{\bar{\gamma}_t}(\bar{\gamma}_{t-1}\bar{\alpha}_{t-1} + \gamma_t \alpha_t)$
8. Set $t \leftarrow t + 1$

Table 1: Pseudo code of the proposed algorithm. In steps 4 and 5, $P_{\Omega_1}(\cdot)$ and $P_{\Omega_2}(\cdot)$ denote the projection to the convex sets Ω_1 and Ω_2, respectively.

Stochastic Gradient Calculation:

Assume y = 1. Then,

$$\hat{F}_t(\mathbf{w}, a, b, \alpha; z) = \hat{p}_t (1 - \hat{p}_t) + (1 - \hat{p}_t) \left(\mathbf{w}^\top x - a\right)^2 - 2(1 + \alpha) (1 - \hat{p}_t) \mathbf{w}^\top x - \hat{p}_t (1 - \hat{p}_t) \alpha$$

$$\frac{\partial \hat{F}_t}{\partial \mathbf{w}} = 2 (1 - \hat{p}_t) \left(\mathbf{w}^\top x - a - 1 - \alpha\right) x$$

$$\frac{\partial \hat{F}_t}{\partial a} = -2 (1 - \hat{p}_t) \left(\mathbf{w}^\top x - a\right), \quad \frac{\partial \hat{F}_t}{\partial b} = 0$$

$$\frac{\partial \hat{F}_t}{\partial \alpha} = -2\hat{p}_t (1 - \hat{p}_t) \alpha - 2 (1 - \hat{p}_t) \mathbf{w}^\top x$$

Assume y = -1. Then,

$$\hat{F}_t(\mathbf{w}, a, b, \alpha; z) = \hat{p}_t (1 - \hat{p}_t) + \hat{p}_t \left(\mathbf{w}^\top x - b\right)^2 + 2(1 + \alpha)\hat{p}_t \mathbf{w}^\top x - \hat{p}_t (1 - \hat{p}_t) \alpha^2$$

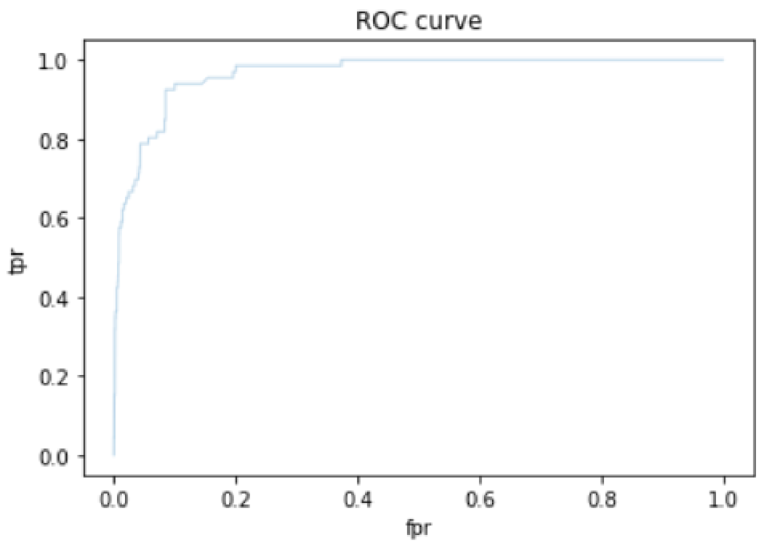
$$\frac{\partial \hat{F}_t}{\partial \mathrm{w}} = 2\hat{p}_t \left(\mathbf{w}^\top x - b + 1 + \alpha\right) x$$

$$. \ \frac{\partial \hat{F}_t}{\partial a} = 0, \quad \frac{\partial \hat{F}_t}{\partial b} = -2\hat{p}_t \left(\mathbf{w}^\top x - b\right)$$

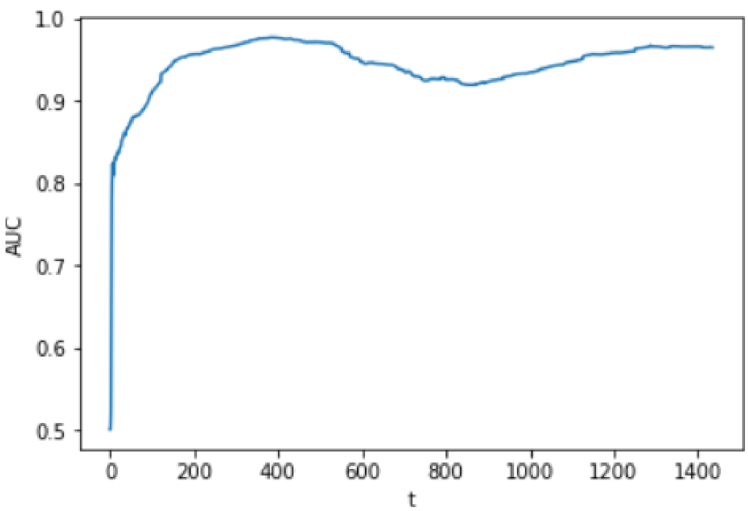
$$\frac{\partial \hat{F}_t}{\partial \alpha} = -2\hat{p}_t (1 - \hat{p}_t) \alpha + 2\hat{p}_t \mathbf{w}^\top x$$

3 Result

I used five fold cross validation to search the learning rate from 2^{-8} to 1 and find learning rate = 0.01844 achieve AUC Score = 0.9346. The ROC curve is shown below:



The AUC score with respect to time t is shown below:



We could achieve AUC score 0. 0.9654 without cross validation.

References:

[1] Ding, Yi, et al. "An Adaptive Gradient Method for Online AUC Maximization." (2015).