# Ticket malicious user identification

**GangTao Zhou, Yuan Gao, Qi Fan**

## Abstract

Based on reference materials[1][2],We use the decision tree algorithm to build the identification strategy for the malicious ticket booking users on the OTA platform. By selecting the relevant features such as user account, browsing and transaction behavior, we use six main features to build the decision tree model, and try to use pruning operation to reduce over fitting. Finally, we try to optimize with RandomForest and GradientBoostring .

## 1 Summary

### 1.1 What kind of problems do we face

There will be some malicious users on the ticket OTA platform. They will use some technical or rule loopholes of the platform to attack and obtain illegal benefits, thus infringing on the rights and interests of normal users and airlines. Their attacks mainly use the platform to protect the rules of normal users, so they are often mixed among normal users. As a platform, we need to identify them in time and set up countermeasures.

### 1.2 How do we solve it

Under the condition of ensuring normal user experience, identifying such malicious users needs to mine the differences between the two types of users. Therefore, we use the existing information of the platform to mine the distinguishing features between the two. At the same time, when identifying malicious users, we need to use the system for immediate intervention to reduce the impact.

### 1.3 What is the idea

However, using traditional manual data analysis, there is a big bottleneck in the timely rate of discovery and processing, because we try to use data visualization methods and build models through machine learning algorithms to realize problem user feature recognition in a systematic way. Considering

## 2 Algorithm Introduction

### 2.1 Entropy

Let x seem to be a discrete random variable with finite values, and its probability distribution is

$$P(X = x_i) = p_i \quad i = 1, 2, \ldots, n \tag{1}$$

The entropy of a random variable is defined as

$$H(X) = -\sum_{i=1}^{n} p_i log p_i \tag{2}$$

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Entropy only depends on the distribution of X and has nothing to do with the value of X, so the entropy of X can also be recorded as $H(P)$, i.e

$$H(p) = -\sum_{i=1}^{\eta} p_i log p_i \tag{3}$$

Conditional entropy H (y|x) is defined as the mathematical expectation of the entropy of the conditional probability distribution of y to x under the given condition of X

$$H(Y|X) = \sum_{i=1}^{n} p_i H(Y|X = x_i) \tag{4}$$

### 2.2 Information gain

$$g(D, A) = H(D) - H(D|A) \tag{5}$$

Let the training data set be D, and |D| represents its sample size, that is, the number of samples. There are k classes $C_k, k = 1, 2, \ldots, K$,$|C_k|$ is the number of samples belonging to $C_k$ ,$\sum_{k=1}^{K} |C_k| = |D|$. Let feature A have n different values $\{a_1, a_2, \ldots, a_n\}$, According to the value D of feature A, it is divided into n subsets $D_1, D_2, \cdots D_n$ is $D_i$ the number of samples, $\sum_{i=1}^{n} |D_i| = |D|$. Record subset $D_i$ belongs to class $C_k$. The set of samples of $C_k$ is $D_{ik}$, i.e. $D_{ik} = D_i \cap C_k$, $|D_{ik}|$ is the Number of samples for $D_{ik}$.

### 2.3 Algorithm of information gain

Input: training data set D and feature A

Output: information gain of feature A to training data set D : $g(D, A)$

1. Calculate the empirical entropy $H(D)$ of dataset D:

$$H(D) = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} log_2 \frac{|C_k|}{|D|} \tag{6}$$

2. Calculate the empirical conditional entropy of feature A to dataset $H(D|A)$

$$H(D|A) = \sum_{i=1}^{n} \frac{|D_i|}{|D|} H(D_i) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \sum_{k=1}^{K} \frac{|D_{ik}|}{|D_i|} log_2 \frac{|D_{ik}|}{|D_i|} \tag{7}$$

3. Calculate information gain

$$g(D, A) = H(D) - H(D|A) \tag{8}$$

### 2.4 information gain ratio

$$g_R(D, A) = \frac{g(D, A)}{H_A(D)} \tag{9}$$

Among them,$H_A(D) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} log_2 \frac{|D_i|}{|D|}$,n is the number of the feature A.

### 2.5 Gini index

In the classification problem, suppose there are K classes, and the probability that the sample points belong to class k is $p_k$. Then the Gini index of the probability distribution is defined as:

$$Gini(p) = \sum_{k=1}^{K} p_k (1 - p_k) = 1 - \sum_{k=1}^{K} p_k^2 \tag{10}$$

For a given sample set D, its Gini index is:

$$Gini\left(D\right)=1-\sum_{k=1}^{K}\left(\frac{|C_k|}{|D|}\right)^2 \tag{11}$$

Here, $C_K$ is the subset of samples belonging to class k in D, and K is the number of classes. If the sample set D is divided into d according to whether the feature a takes a possible value, a is divided into $D_1$ and $D_2$ two parts,

$$D_1=\left\{(x,y)\in D|\,A\left(x\right)=a\right\},D_2=D-D_1 \tag{12}$$

Then, under the condition of feature A, the Gini index of set D is defined as

$$Gini\left(D,A\right)=\frac{|D1|}{|D|}Gini\left(D_1\right)+\frac{|D_2|}{|D|}Gini\left(D_2\right) \tag{13}$$

$Gini\left(D\right)$ represents the uncertainty of set D, and $Gini\left(D,a\right)$ represents the uncertainty of set D after A = a segmentation.

### 2.6 CART generation algorithm

Input: training data set D, conditions for stopping calculation

Output: CART decision tree

1. According to the training data set, starting from the root node, recursively perform the following operations on each node to build a binary decision tree:Let the training data set of several points be D, and calculate the Gini index of the existing features to the data set. At this time, for each feature a, for each possible value a, D is divided into d according to the test of sample point A = a as "yes" or "no" $D_1$ and $D_2$ two parts, calculate the Gini index when A = a.
2. Among all possible features A and all their possible cut points a, the feature with the smallest Gini index and its corresponding cut point are selected as the optimal feature and optimal cut point.According to the optimal feature and optimal cut point, two sub nodes are generated from the current node, and the training data set is allocated to the two sub nodes according to the feature.
3. Call 1 and 2 recursively for two child nodes until the stop condition is met.
4. Generate CART decision tree

The condition for the algorithm to stop calculation is that the number of samples in the node is less than the reservation threshold, or the Gini index of the sample set is less than the reservation threshold (the samples basically belong to the same class), or there are no more characteristics.

## 3 Related Work

### 3.1 Algorithm selection

We need to take different algorithms depending on whether there are category labeled columns or not, if there are labeled columns we use decision tree algorithm and use decision tree to do supervised classification. Using labeled problem users to calculate the identifying features of these problem users. Finally tree visualization and feature visualization are used to get an overall picture of the at-risk users.

### 3.2 Data Structure

User data.

Account information (account number, registration time, registration location, historical consumption data) Browsing data (order number, page dwell time, device ID, IP address)

Order data.

Order data (account number, order number, order placement time, order status, flight number, flight date, departure city, arrival city) Passenger information (order number, passenger name, document type, document number) Flight information (flight number, flight date, turnover rate) ER relationship please see 1.

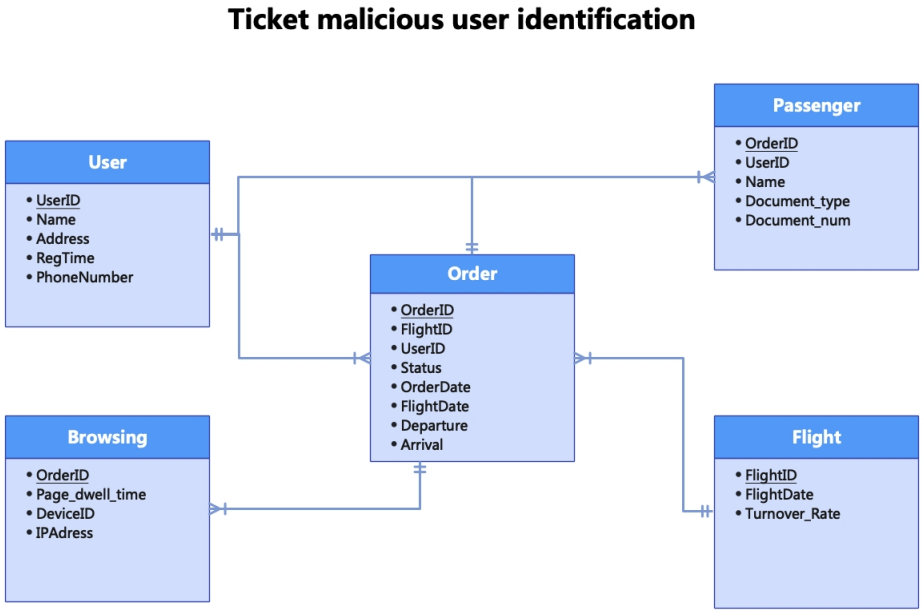


Figure 1: ER

### 3.3 Analyzing Data

We have selected six characteristics from the order and account dimensions that may help us identify malicious users. By looking at the data, we can identify outliers or problems in the data in advance. Data Analysis Results please see 2.

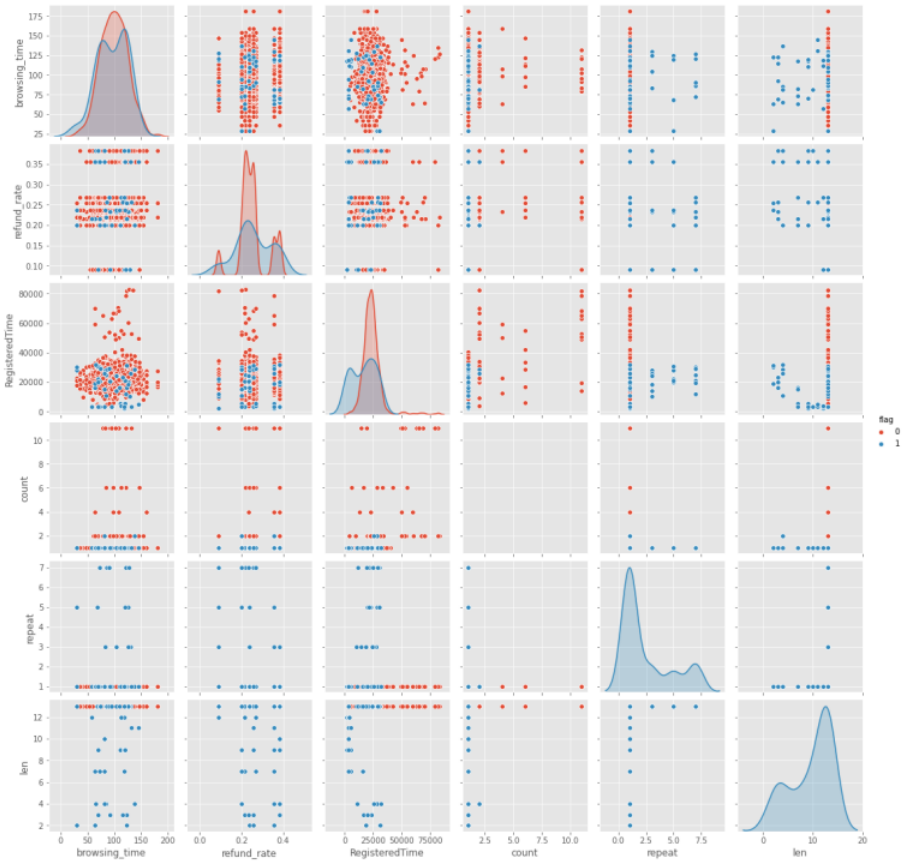


Figure 2: DR

### 3.4 Model training

In this phase, we need to put together the prepared algorithm and the processed data set after data feature engineering, and the algorithm object calls the fit method to pass the training set into.

Method used please see Split Data and DecisionTree .

### 3.5 Evaluation model

In this phase, the main thing to do is to check the category label columns of the predicted columns in the predicted dataset against the original category label columns to check the goodness of the model. The bottom layer here is to compare the data in the two columns, and then the accuracy of the prediction is also rated the good or bad model. Using score(), the data and labels of the test sample are input and the classification score of the model is returned after the test sample prediction.

Method used please see:Model Evaluation

For evaluation results, please see ConfusionMatrix 3.

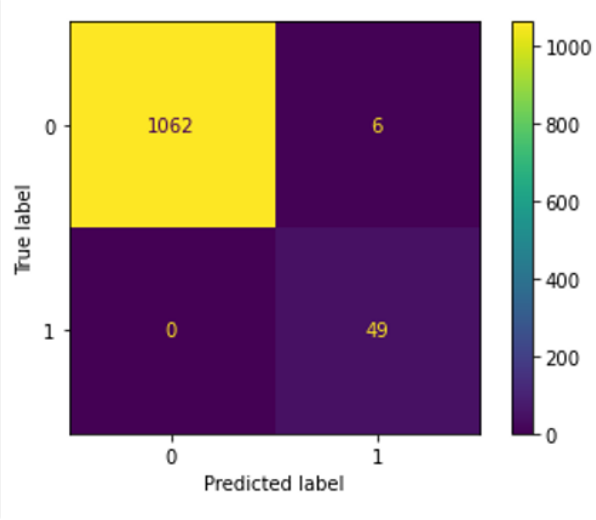


Figure 3: Confusion Matrix

### 3.6 Analysis Tree

Through pre pruning and post pruning, the complexity of decision tree can be controlled to reduce the risk of over fitting. We can control the depth of the tree and limit the number of leaf nodes. please see Feature Importance 4 , Tree Graph 5 and Tree (pre-pruning) 6.

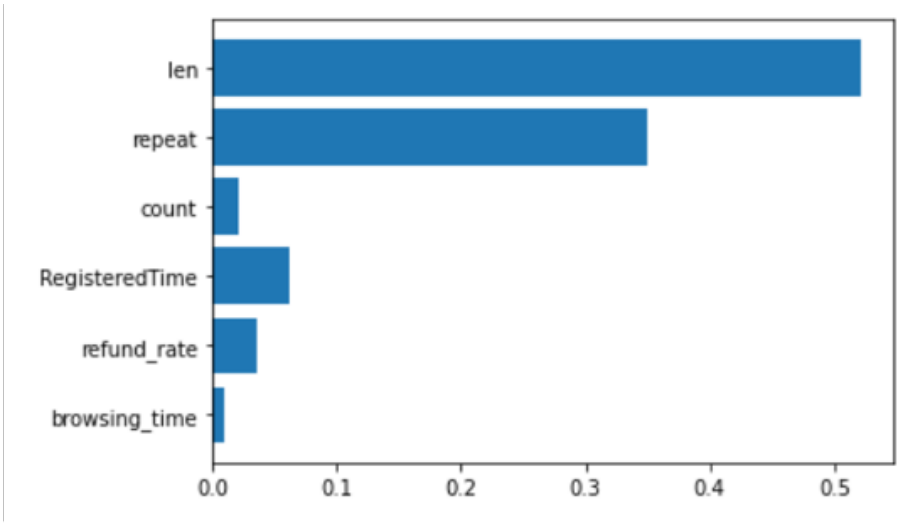


Figure 4: Feature Importance

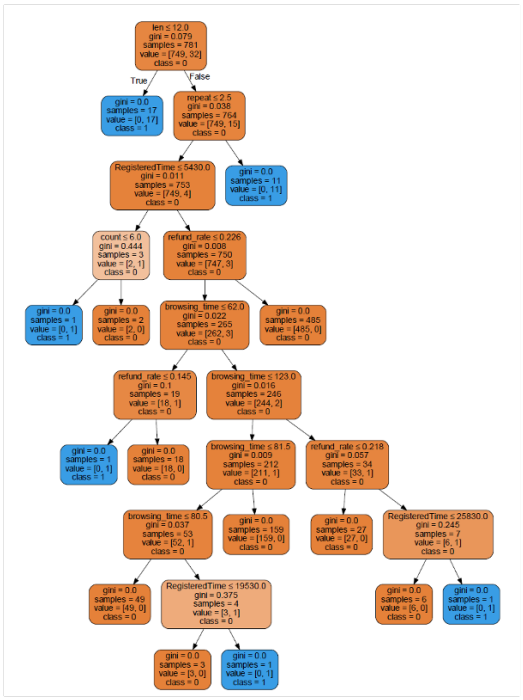


Figure 5: Tree Graph

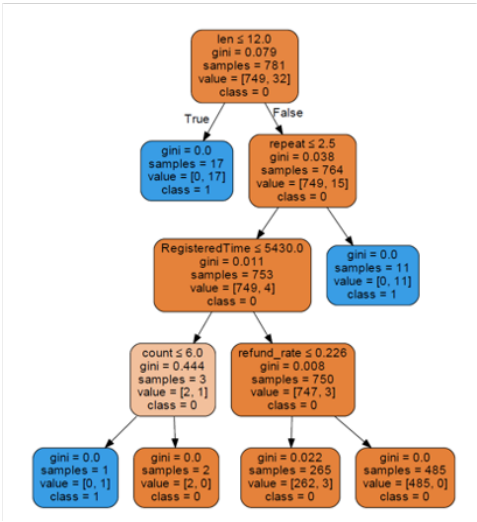


Figure 6: Tree (pre-pruning)

## 4 Problem Setting

Consider using decision trees to do supervised classification, we need to effectively identify this part of the malicious users, through the algorithm out of this part of the problem users exist to identify the characteristics, through pre-pruning (pre-pruning), post-pruning (post-pruning) control the complexity of the decision tree to reduce the risk of overfitting, and finally through the tree visualization and feature visualization to understand the overall situation of the risk users, to reduce the damage caused by these malicious behavior.

## 5 Optimization Scheme

The Decision Tree is simple, logical and interpretable, but it also has a big disadvantage: very easy to over fit. Random Forest can improve this problem. It integrates many decision trees to form a more powerful tree model. Random Forest is a Bagging method. Bagging is a sampling method with put back: take a sample and add it to the training set, and then put the sample back to the original sample space, so that the sample can still be taken next time. In this way, multiple training sets are obtained, and each training set is different. On the basis of bagging, Random Forest further introduces some randomness into the training process of decision tree: when deciding to divide attributes, first randomly select a subset containing K attributes, and then select an optimal attribute from the subset for division, which is also the connotation of random. Random forest is simple, easy to implement, low computational overhead, and performs well in practical tasks. However, there is no connection between the trees in the Fandom Forest. We can improve this by using GBDT. GBDT will take the residual value of the preamble tree as the learning objective to fit. The final output of the model is to add the results of all trees and optimize the loss function with the gradient method. In addition, XGBoost goes further on the basis of GBDT and adds a regular term to the objective function of each iteration to further reduce the risk of over fitting. This is the result:

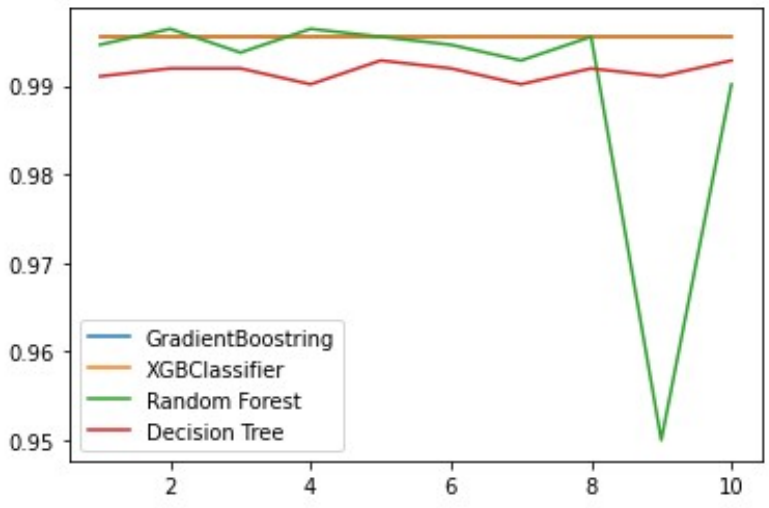


Figure 7: Results

## 6 Code

Please visit the following link to get the code:

https://github.com/kidzss/ML584Project.git

## References

[1] Li hang. Statistical learning method. *ISBN*, 978-7-302-51727-6.

[2] Andreas C.Muller,Sarah Guido. Introduction to Machine Learning with Python. *ISBN*, 978-7-115-47561-9.

[3] Olshen R A,Quinlan J R. Induction of decision tresss.Machine Learning. 1986,1(1):81-106.

[4] Olshen R A,Quinlan J R. programs for machine learning. Morgan Kaufmann,1992.

[5] Liu B. Web data mining:exploring hyperlinks,contents and usage data. Springer-Verlag,2006.

[6] Ripley B. Pattern recognition and neural networks. Cambridge University Press,1996.