ORIGINAL ARTICLE

An XGBoost-based casualty prediction method for terrorist attacks

Yi Feng[1] · Dujuan Wang[1,2] · Yunqiang Yin[3] · Zhiwu Li[4,5] · Zhineng Hu[1]

Abstract

Terrorist attacks have been becoming one of the severe threats to national public security and world peace. Ascertaining whether the behaviors of terrorist attacks will threaten the lives of innocent people is vital in dealing with terrorist attacks, which has a profound impact on the resource optimization configuration. For this purpose, we propose an XGBoost-based casualty prediction algorithm, namely RP-GA-XGBoost, to predict whether terrorist attacks will cause the casualties of innocent civilians. In the proposed RP-GA-XGBoost algorithm, a novel method that incorporates random forest (RF) and principal component analysis (PCA) is devised for selecting features, and a genetic algorithm is used to tune the hyperparameters of XGBoost. The proposed method is evaluated on the public dataset (Global Terrorism Database, GTD) and the terrorist attack dataset in China. Experimental results demonstrate that the proposed algorithm achieves area under curve (AUC) of 87.00%, and accuracy of 86.33% for the public dataset, and sensitivity of 94.00%, AUC of 94.90% for the terrorist attack dataset in China, which proves the superiority and higher generalization ability of the proposed algorithm. Our study, to the best of our knowledge, is the first to apply machine learning in the management of terrorist attacks, which can provide early warning and decision support information for terrorist attack management.

Keywords Terrorist attack · Prediction · Extreme gradient boosting · Feature selection · Area under curve

Introduction

Global Terrorism Database defines terrorist attacks as non-state actors that threaten or actually use illegal force and violence to achieve political, economic, religious or social goals through fear, coercion or intimidation (https://www.start.umd.edu/gtd). All terrorist incidents contain three attributes: (i) the incident must be intentional; (ii) the incident must cause a degree of violence or a direct threat of violence; and (iii) the perpetrator of the incident must be sub-national actors (https://www.start.umd.edu/gtd). Many researches emphasize that the purpose of terrorist attacks lies not in the violent act itself, but in furthering specific political, religious and other goal [4, 21]. The occurrence of an incident will not only bring death and injury to innocent people, but also cause infrastructure damage and social panic. According to the statistics of the global terrorism dataset, from 1970 to 2017, the average number of terrorist attacks occurred 3800 per year. Between 2005 and 2015, the number of terrorist attacks increased significantly year by year, and its number has steadily declined since 2015. In 2014, the number of terrorist attacks occurred as high as 16,000. Since September 11, 2001, 4000 people on average have been killed in terrorist attacks each year. Terrorism emerged as a global threat following the attacks on September 11th. Although the intention of terrorist attacks is to generate widespread fear, not to bring about any substantial harm [30, 54], it is invariably necessary to study whether the terrorist attacks will bring life threats to innocent people. Since the management and control of terrorist attacks are exceedingly complicated, it is important to develop efficient methods to predict casualties in terrorist attacks.

✉ Yunqiang Yin
yinyq@uestc.edu.cn

1 Business School, Sichuan University, Chengdu 610064, China

2 Institute of Economy and Enterprise Development, Sichuan University, Chengdu 610064, China

3 School of Management and Economics, University of Electronic Science and Technology of China, Chengdu 611731, China

4 School of Electro-Mechanical Engineering, Xidian University, Xi'an 710071, China

5 Institute of Systems Engineering, Macau University of Science and Technology, Taipa, Macau SAR

Machine learning is a kind of algorithm (method) that parses data, learns from it, and then makes decisions or predictions about something in the world. In recent years, as a powerful tool for the decision support in emergency management, machine learning algorithms have been applied for studying the casualties caused by natural disasters, such as earthquakes [52, 53]. However, to date, machine learning algorithms have been rarely used in predicting the casualties of terrorist attacks. To fill this gap, we propose an XGBoost-based casualty prediction algorithm namely RP-GA-XGBoost to study whether terrorist attacks will lead to the casualties of innocent civilians. The results of this paper will not only assist the decision makers in rapidly allocating emergency resources and implementing the emergency plans, but also provide important reference values for the governments' decision-making.

In the proposed algorithm RP-GA-XGBoost, we devise a hybrid feature selection method that incorporates random forest (RF) and principal component analysis (PCA), which is more advantageous than other feature selection methods as demonstrated by experiments. Meanwhile, a genetic algorithm is used to tune the hyperparameters of XGBoost. Based on Global Terrorism Database namely GTD, we compare the RP-GA-XGBoost algorithm with several state-of-the-art methods, such as Logistic Regression (LR), Adaboost, XGBoost, RF, decision tree (DT), support vector machine (SVM), in terms of performance and accuracy. The experimental results demonstrate that RP-GA-XGBoost significantly outperforms the aforementioned algorithms. We also conduct experiments on terrorist attacks in regions. Taking China as an example, the performance of RP-GA-XGBoost has been greatly enhanced because the problem is more specific and targeted in a domestic setting. Therefore, one can conclude that RP-GA-XGBoost is an efficient and effective algorithm for terrorist's incident management in specific scenarios.

The rest of this paper is organized as follows: In "Literature review" we briefly review the related literature to position our research. In "Casualty prediction method for terrorist attacks", we introduce the initial database, and describe the process of the whole experiment based on our proposed RP-GA-XGBoost in details. In "Experimental results", we conduct extensive numerical experiments to evaluate the performance of the proposed method. Some conclusions are drawn and topics for future research are suggested in "Conclusion".

Literature review

The quantitative methods used for studying the terrorist attacks are divided into two classes: (i) statistical methods, and (ii) machine learning methods. In the following we first give a brief literature review that use statistical and machine learning methods for studying the casualties of terrorist attacks, and then pose our research method and provide the main contributions of this paper.

In the early years, statistics as an effective data analysis method is usually used by scholars to analyze terrorist attack data. Mohler et al. [33] and Clauset et al. [8] propose a general statistical estimation algorithm that combines a semi-parametric model of tail behavior and a nonparametric bootstrap to predict the future probability of a large terrorist attack based on historical data on terrorist attacks. LaFree et al. [24] describe the collection of the dataset, discuss the pros and cons of general open source data, and conduct descriptive statistical analysis. Borooah et al. [5] carry out a study on terrorist attacks in India based on the dataset between 1998 and 2004, and perform a statistical analysis of the perpetrators and modus operandi of terrorist incidents to explore the extent to which the number of deaths is affected by the type of attack and different terrorist attack groups. Arnold et al. [3] compare the casualties and other outcomes caused by different types of explosions. Note that, however, the above statistics-based approaches focus mainly on searching, organizing, and describing data with no predictive results being involved. As an extension of statistical learning in practice, machine learning provides algorithm technical support for practical problems.

Machine learning is one of the popular techniques of emergency management and decision-making, which can effectively acquire and disseminate real-time disaster information [59]. Classification is one of the core issues in data mining task, which reflects the association between features and classified labels [18, 59]. Recently, machine learning has been widely used to solve the classification problems on the terrorist attacks.

Several studies use traditional classifiers or simple models to explore and predict terror-related issues. Fahey et al. [13] study the difference between terrorist air hijacking and non-terrorist air hijacking. First, they divide 1019 air hijackings that occurred worldwide from 1948 to 2007 into terrorist incidents and non-terrorist incidents according to the definition of terrorism. After that, they predict whether air hijacking aims at terrorist attacks by using LR analysis. The results demonstrate that organization resources, public resources and publicity can effectively distinguish whether an air hijacking is a terrorist. Due to the scale and complexity of the data, it is difficult to determine the patterns and trends of terrorists. For this, Sachan and Roy [38] establish a terrorist group prediction model (TGPM), and predict terrorist incidents by learning the means and other information of terrorist group attacks from various terrorist attacks. Enders et al. [12] design a time series method of dividing terrorism into transnational and domestic terrorist events. Based on calibrated data, the authors explore the dynamic

relationship and correlation between domestic and transnational terrorist events, and find that there is a significant simultaneous and lagging cross-correlation between domestic and transnational terrorist events. We can observe that the above studies are more about the exploration and prediction of terror-related issues by using traditional classifiers or simple learning devices.

Hybrid models are also proposed to solve predicted issues related to terrorist attacks. Zhang and Mahadevan [58] develop a hybrid model that combines neural network and SVM to quantify the risk level of aviation event anomaly, by which one can predict the severity of an aviation event. Meng et al. [31] put forward a hybrid classifier which uses SVM, K-nearest neighbor (KNN), Bagging and C4.5 to predict the types of terrorist attacks, and use a genetic algorithm to determine the weight of each model. Shafiq et al. [40] propose a hybrid classifier that includes KNN, Naïve Bayes, DT to predict the types of terrorist attacks. Khorshid et al. [20] predict which terrorist organizations should be responsible for attacks in the Middle East and North Africa using SVM, DT and Naïve Bayes. Soliman and Tolan [43] use Naïve Bayes, SVM and DT to predict terrorism organizations in Egyptian terrorist attacks from 1970 to 2013.

Based on the above literature review, we find that there are few research results on the casualties of terrorist attacks using machine learning methods. However, prediction of casualties and demand using machine learning methods has been investigated in the field of natural disasters. For example, Xu et al. [53] develop a hybrid prediction method based on empirical mode decomposition (EMD) and autoregressive integrated moving average model (ARIMA) to predict the commodity demand after natural disasters. Wang [50] uses a genetic algorithm based on GM (1,1) and Fourier series to predict food demand after snow disasters. Wang et al. [48] build a back propagation (BP) neutral network model to forecast earthquake casualties, by considering earthquake magnitude, depth of hypocenter, intensity of epicenter, level of preparedness, earthquake acceleration, population density, and disaster forecasting as the experimental features. Dogan and Akgungor [10] use nonlinear multiple regression (NLMR) and artificial neural network (ANN) methods to predict road injuries in Turkey.

Obviously, the factors affecting casualties caused by natural disasters and terrorist attacks are different. Disaster grades and building quality are the main influencing factors of casualties caused by natural disasters, and the information on these two factors can generally be quickly obtained through geological monitoring, weather warning and field investigations. However, terrorist attacks are man-made disasters. They are commonly caused by human beings. The casualties of terrorist attacks are not only directly related to natural factors (weather, geographical location, etc.), but also human factors (number of perpetrators and their

psychological mindset). Human factors make the casualty prediction of terrorist attacks more complicated and difficult than that of natural disasters. As a consequence, it is necessary to devise more efficient machine learning methods to predict the casualties in terrorist attacks. Motivated by this, this paper aims at developing a novel machine learning method to predict the casualties in terrorist attacks so as to provide early warning and decision support information for terrorist attack management.

At present, many studies have found that two factors directly affect the performance of machine learning methods: the value of hyperparameters and the selection of features [18, 31, 42, 53, 56].

- For the determination of the optimal value of hyperparameters, the commonly used methods are manual search, grid search algorithm, and Bayesian hyperparameter optimization [31, 42, 53, 56]. For grid search algorithm, since the classification accuracy corresponding to the parameter groups in the grid is greatly low, only the parameter groups in the smaller interval have higher classification performance. Therefore, it is quite time-consuming to train all the points in the grid to find the best hyperparameters. Compared with manual search method and grid search algorithm, Bayesian hyperparameter optimization have fewer iterations, faster speed [51]. In recent year, many researchers use intelligent algorithm such as genetic algorithm (GA), particle swarm optimization (PSO) for hyperparameter optimization to further improve the performance of the model [9, 27]. Compared with PSO, GA can search for the final hyperparameters faster and improve the accuracy [45]. Many researchers used genetic algorithms to study the hyperparameter optimization of RF [1], support vector machine [9, 45] and artificial neural network [2], respectively, and the experimental results show that the performance of the proposed methods are better than traditional machine learning methods. However, XGBoost, as an advanced machine learning method, has rarely been studied on its hyperparameter optimization.
- For the selection of features, the commonly used feature selection methods are filter, embedded and wrapper methods [37]. Filtering methods are generally used as a preprocessing step, which is a part of statistics that is independent of machine learning, and selects features by means of scores on statistical experiments and correlation indicators. Embedded methods allow algorithms to decide which features to use, that is, feature selection and algorithm training are carried out simultaneously. Wrapper methods use the final classifier as the evaluation function of the feature selection, and select the optimal feature subset through the classification results. However, wrapper methods train the model repeatedly, which

leads to expensive computational cost and high overfitting risk [49]. Meanwhile, in these three feature selection methods, the number of features is commonly determined subjectively by the researchers in many research [31, 56], leading to low accuracy and poor performance of the model.

Compared with the aforementioned studies, the main contributions of this paper are as follows.

First, we propose an XGBoost-based casualty prediction algorithm RP-GA-XGBoost based on data mining and optimization algorithms to study whether terrorist organization's behaviors will cause innocent casualties. With the powerful ability to control the complexity of the model, XGBoost-related methods have been extensive applied in various fields, such as financial transaction [34], physical fitness evaluation [15], medical prediction [26] and so on. What is more, 17 of the 29 award-winning solutions were obtained by using XGBoost in the 2015 machine Learning Kaggle Competition [56]. However, XGBoost has not been applied in the research of terrorist attacks. Motivated by this, we use XGBoost incorporating with some enhancement strategies to predict whether terrorist attacks will cause the casualties of innocent civilians.

Second, we apply a hybrid feature selection method for selecting features, which incorporate the RF and PCA to makes up for the shortcomings of feature selection in references [31, 56], and it can effectively reduce the computational cost. In this method, the important scores of each feature are obtained through RF, and the number of features is determined by mean of the cumulative explained variance contribution rate curve. Finally, feature subset is obtained according to the rank of feature important scores and the number of features.

Third, we apply intelligent optimization algorithm to tune the hyperparameters of XGBoost. Due to XGBoost has a large number of hyperparameters, and thus the selection of the most appropriate hyperparameters can effectively the accuracy of casualty prediction for terrorist attacks we formulate. Based on above observation [1, 2, 9, 41], the paper chooses genetic algorithms to tune the hyperparameters of XGBoost.

Casualty prediction method for terrorist attacks

In this paper, we develop XGBoost-based casualty prediction algorithm, namely RP-GA-XGBoost, to predict whether the terrorist organization's attack will lead to casualties of innocent people. Figure 1 shows the construction process of the prediction method. First, we deal with missing values, features and labels in the initial dataset, and divide the whole

experimental dataset into training set and test set. Second, we use a new hybrid feature selection method RF-PCA to select the experimental features. After that, the hyperparameters of XGBoost classifier are tuned using a genetic algorithm (GA-XGBoost). Finally, we use evaluation indicators to evaluate the trained model on the test set. The details of the main steps are described in the following sections.

Data preprocessing

For data mining and machine learning, data preprocessing often accounts for about 70% of the workload, and the quality of the data set determines the success of the experiment [32]. Before constructing the model, plenty of time and efforts are spent in processing the data to ensure the validity of the results.

The data used in this paper is derived from the Global Terrorism Database (GTD), established by the University of Maryland and the US Department of Homeland Security, and maintained by the National Terrorism and Counter-Terrorism Research Alliance (START) (https://www.start.umd.edu/gtd). The Global Terrorism Database (GTD) is an open database; all the information in the data comes entirely from public open source materials, including news, books, existing data sets, and legal documents. This data includes information on global terrorism events from 1970 to 2017. The GTD contains 135 features and more than 100,000 samples, but not all of the data is used directly in this paper.

Since we focus only on whether the behavior of terrorist organization will lead to innocent casualties, the casualties caused by non-terrorist attacks are beyond the scope of this paper. Only data that meets the following three criteria is included: (1) The attack is conducted on the basis of political, socioeconomic or religious motives; (2) The incident is based on coercion, intimidation or intent to publicize to wider audiences, and (3) The attack exceed the scope of international humanitarian law. At the same time, in order to eliminate some suspicious events such as guerrilla warfare, internal conflicts and mass murder, we only consider the data collected since 1998. It is important to note that the casualties considered here do not include the casualties of terrorists, but only the casualties of innocent civilians.

Missing value and feature processing

It is worth noting that the selected data based on the above criteria is not complete, which has a great impact on the training of the model. For this, we first remove the records that have null values or abnormal values, and then remove redundant features and the nominal features that have multitudinous values or missing values. For example, for the attributes "location", "alternative", "attacktype2", "attacktype3", "targsubtype2", and "natlty2", 70% samples contain

Fig. 1 A flowchart of the proposed method
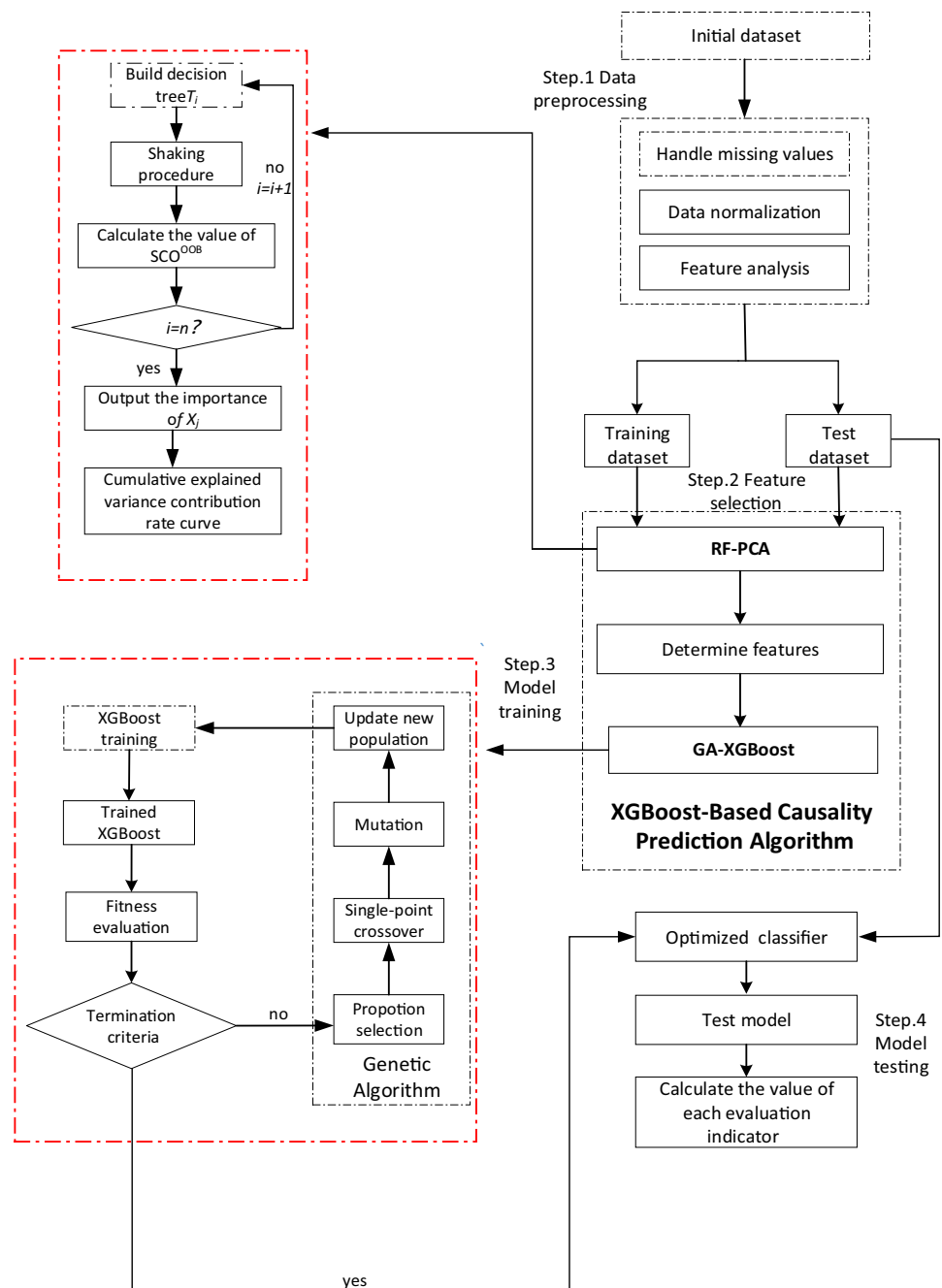
missing values. After preprocessing, we obtain 87,773 samples and 32 features, summarized in Table 12.

For the chosen samples, we denote by "nkill" the number of deaths, and "nwound" the number of injured. We define "ncasualty" as a binary label equal to 1 if and only if the sum of the deaths toll and the number of injuries is greater than 0, indicating that terrorist attacks can lead to innocent casualties. Among the 87,773 samples, there are 58,261 samples that cause innocent casualties, and 29,512 samples that do not cause innocent casualties. Based on reference [23], we divide the dataset into two parts, in which 70% of the data

is used for the training set and other 30% is used for the testing set.

One-hot code

XGBoost is only suitable for processing numeric vectors, it is necessary to convert other forms of data into numeric vectors. For the considered problem, we need to convert the feature attributes into numerical values. To this end, we use the one-hot encoding method, which is a process by which feature

attributes are converted into a form suitable for machine learning algorithms to do a better job in prediction.

To be precise, suppose that there are n samples, each of which has a discrete feature with m attribute values. The one-hot code uses the *m*-bit register to binary encode the state of *m* attribute values, and for each sample, only one bit is active at any time. By one-hot encoding each sample, this feature is transformed into a sparse n*m matrix. In this paper, we use the one-hot code to convert the data types of the five features, say "country_txt", "attacktype1_txt", "weapsubtype1_txt", "targtype1_txt", and "natlty1_txt", into numeric data type. Taking "attacktype1_txt" as an example, the categories of attack types include assassination, hijacking, kidnapping, barricade incident, bombing/explosion, armed assault, unarmed assault, facility/infrastructure attack and unknown. Suppose that there are five samples now. The ways of attack are hijacking, barricade incident, armed assault, kidnapping, and facility/infrastructure attack, respectively. In terms of the one-hot code, a sparse matrix depicted in Fig. 2 is obtained.

Data normalization

In machine learning practice, we commonly need to convert different specifications of data into uniform specifications, or to convert differently distributed data into specific distribution of requirements, i.e., the data needs to be dimensionless. In machine learning and data mining, although many algorithms (LR, DT) do not entail normalized data, Le et al. [25] point out that feature re-coding and data normalization not only enhance the performance of the classifier, but also speed up the solving speed and improve the quality of the model. Min–max standard is the most commonly used method in normalization, which has been extensively applied to eliminate dimension effects [39, 57]. Therefore, we use the min–max standard method to normalize the data before building the model, which converts the data values into the range of [0,1] [57]. The conversion formula is defined as follows:

$$x_i'(k) = \frac{x_i(k) - \min_i x_i(k)}{\max_i x_i(k) - \min_i x_i(k)}, \tag{1}$$

where k and i represent the feature and the sample, respectively, and $x_i(k)$ denotes the k-th feature of the i-th sample.

$$\begin{array}{c} \text{Sample1} \\ \text{Sample2} \\ \text{Sample3} \\ \text{Sample4} \\ \text{Sample5} \end{array} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Fig. 2 Sparse matrix of "attacktype1_txt"

XGBoost-based casualty prediction algorithm

Feature selection method based on RF-PCA

In machine learning, the results of feature selection are directly related to the performance of the model [16, 37]. Feature selection method reduces the computational cost and improves the model performance by reducing the feature dimension and data scale. Since not all features contribute to the construction of the model, the redundant features are removed [16, 37]. For feature selection method filtering, since the evaluation indicators of features the are independent of the specific learning algorithm in filtering algorithm, the selected feature subset does not perform well in the classification model. Given the flaws of the general feature selection methods and the uncertainty of the number of features, this paper proposes a hybrid feature selection method called RF-PCA to determine experiment features. RF-PCA measure the importance of features based on the OOB error rate, and uses cumulative explained variance indicator of PCA to determine the number of features. RF-PCA will be described in detail below.

Random forest (RF) is a typical bagging ensemble machine learning method proposed by Breidman in 2001 [55]. Each classifier operates independently and in parallel with each other. There is no dependency between each classifier, and the results of all classifiers are finally summarized [6]. It uses bootstrap and node random splitting techniques to construct multiple decision trees. These decision trees are trained independently and each tree outputs its own results, and final result is a voting or averaging method that yields the final result [55]. RF is commonly used for classification analysis or regression analysis, but it also has good performance in feature selection and is robust to noise data [19, 22, 44, 47]. In RF, the feature importance is mainly measured by the Gini index method and the OOB error rate. When the types of variable are different in the data, the performance of the Gini index is worse than the OOB error, and the OOB error rate is more widely used than the Gini index in existing studies [22]; therefore we adopt the OOB error rate in this experiment. In what follows, we will introduce the OOB error rate in detail.

Since the generation process of decision tree in RF uses bootstrap, not all samples are used in the generation of a tree, which refer to the unused samples as out of bag. The OOB error rate is the difference between the classification accuracy of the data outside the bag before and after slight disturbance. To be precise, let D_b be the training samples set, b ($b = 1, 2, \ldots, B$) random sampling times, and $X_j(j = 1, 2, \ldots, N)$ the j-th feature. The variable importance measure based on the classification accuracy of feature X_j can be calculated as follows: $\text{Score}_j = \frac{1}{B} \sum_{b=1}^{B} \left(Z_b^{\text{oob}} - Z_{bj}^{\text{oob}} \right).$

First, decision tree T_1 is created on the training set, and the data out of the bag is denoted as L_1^{oob}. Then, decision tree $T_b\,(b=1,2,\ldots,B)$ classifies the data out of bag and the number of correct classifications is recorded as Z_b^{oob}. After that, we randomly change the value of the feature $X_j\,(j=1,2,\ldots,N)$ in the L_b^{oob} by noise, that is, $X_j = X_j + noise\,(j=1,2,\ldots,N)$, and denote by L_{bj}^{oob} the data after permute. The new data L_{bj}^{oob} is classified using T_b again, and the number of correct classifications is denoted as Z_{bj}^{oob}.

After generating the feature importance ranking, we obtain the importance score of each feature in all features. The next step of our method is to determine is to determine the number of features through PCA to achieve the optimal model. To achieve this, PCA is used to transform the feature space into new feature space to obtain the interpretable variance contribution of features. Then, we draw the cumulative explained variance contribution rate curve to determine the number of feature selections. By plotting the cumulative explained variance contribution rate curve, where the abscissa is the number of features and the ordinate is the cumulative explained variance contribution rate, we can observe the relationship between the number of features and the cumulative explained variance contribution rate. As the number of features increases, the slope gradually slows down and the cumulative explained variance contribution rate approaches 1. Therefore, if the cumulative explained variance contribution rate curve tends to be stable when the number of features is T, that is, with the increase in the number of features, the cumulative explained variance contribution curve has no obvious growth trend, then we set the value of the number of features as T. Finally, the feature subset is obtained according to the order of the feature importance scores and the determined number of features

selected. Table 1 illustrates the pseudo-code of our proposed feature selection method (RP) that combines RF and PCA.

Figure 3 displays the feature scores based on random forest OOB rate in descending order, where the abscissa represents the feature name and the ordinate denotes the score of the corresponding feature. From Fig. 3, we observe that the time variables (iyear, imonth, iday) dominate the others. However, in the casualty prediction of future terrorist attacks, we attempt to ensure model that learns from the behavior and characteristics of terrorists to make predictions, rather than model that relies on time variables (iyear, imonth, iday) to make predictions. Note that in some existing studies about prediction based on GTD [31, 38, 40], the researchers also does not consider time variables in the model construction process. Thus, this paper does not
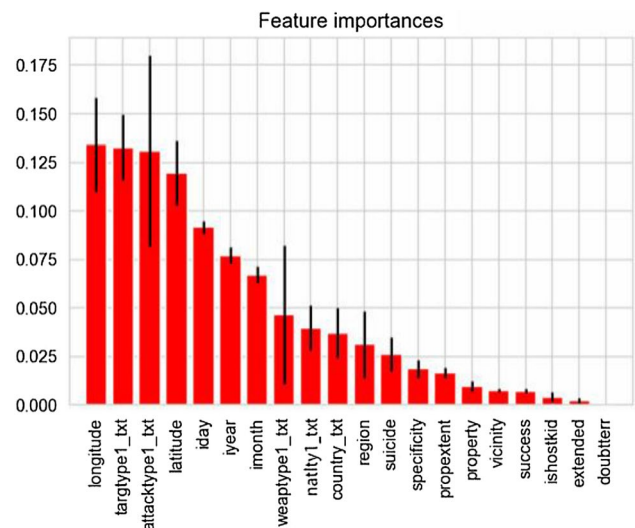

Fig. 3 Importance feature scores by random forest

Table 1 Process of the RF-PCA feature selection mechanism

RF-PCA feature selection
Input: The training dataset $D = (x_i, y_i), (i=1,2,3,\ldots,m), x_i \in R^N$, where N is the number of features, $y_i \in \{0,1\}$, and the feature set is $\{X_1, X_2, \ldots, X_N\}$
Output: The set of experiment features
1. For $b=1$ to B do
2. Generate D_b from D through bootstrap as training subset, and generate out of bag L_b^{oob} as testing subset
3. Train decision tree T_b on the training subset D_b
4. Use T_b to predict on the testing subset L_b^{oob}, and calculate the number of correctly classified samples Z_b^{oob}
5. For $j=1$ to N do
6. Permute the values of X_j in L_b^{oob}, and generate permutation dataset L_{bj}^{oob}
7. Use T_b to predict on L_{bj}^{oob}, and calculate the number of correctly classified samples Z_{bj}^{oob}
8. End for
9. End for
10. Calculate the importance score S_j of the feature X_j, with $S_j = \frac{1}{B}\sum_{b=1}^{B}(Z_b^{oob} - Z_{bj}^{oob})$
11. Generate feature importance set $S = \{S_1, S_2, \ldots, S_n\}$, where the items are sequenced in a descending order
12. Plot explained cumulative variance, and determine the number of features T
13. Generate the set of experiment features $\{S_1, S_2, \ldots, S_T\}$

involves the time variables in the model construction process. Figure 4 shows the relationship between the number of features and the cumulative explained variance, where the abscissa is the number of features and the ordinate is the cumulative explained variance contribution rate. According to the PCA analysis, we observe that when the number of features reaches nine, the cumulative explained variance curve has no obvious growth trend. Therefore, we select the first nine features i.e., "_txt", "attacktype1_txt", "longitude", "latitude", "weaptype1_txt", "natlty1_txt", "country_txt", "region", and "suicide"; these features will be used for model training and testing.

Due to learning independence, filtering methods generally have better generalization ability, and its computational cost is less than wrapper methods and embedded methods [36]. To assess the efficiency of RF-PCA, we compare it with Chi-square filtering, which is a commonly used method of feature selection for classification problems [23]. Figure 5

shows the results of using Chi-square to filtering features. The first nine features are selected for this experiment. By comparing experiments using different features for model training, we demonstrate that RF-PCA has better performance than Chi-square filtering when using the same number of features for model training and testing.

Hyperparameter tuning method based on GA-XGBoost

The core of ensemble learning is to construct several different models and aggregate them to improve the performance of the models. According to the integration strategy of different structures, the ensemble learning method can be divided into two categories: One is the parallel method, which is also called bootstrap aggregating (bagging), and the other is the sequential method [11]. XGBoost-based casualty prediction algorithm proposed in this paper belongs to the second category. Figure 6 shows the flow chart of sequential
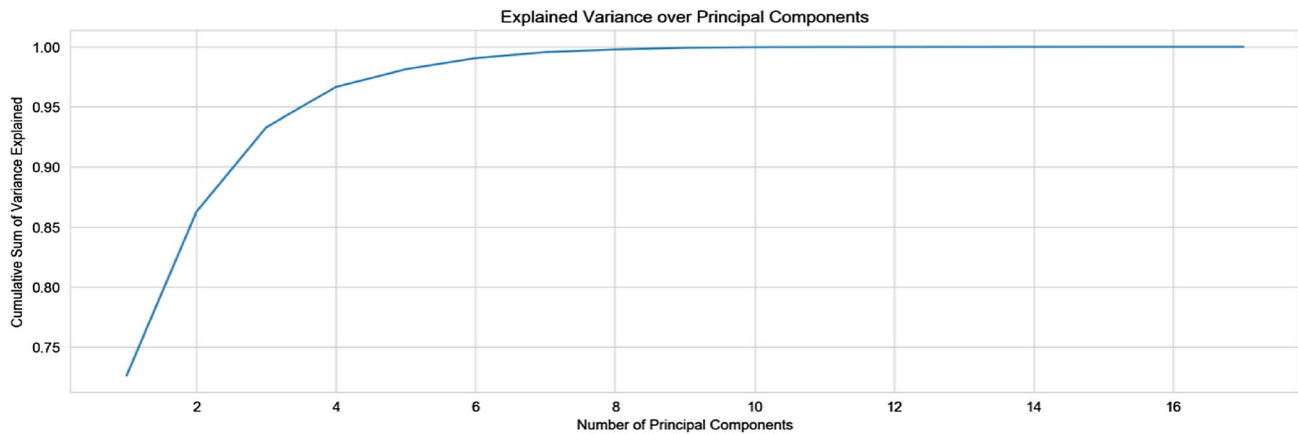

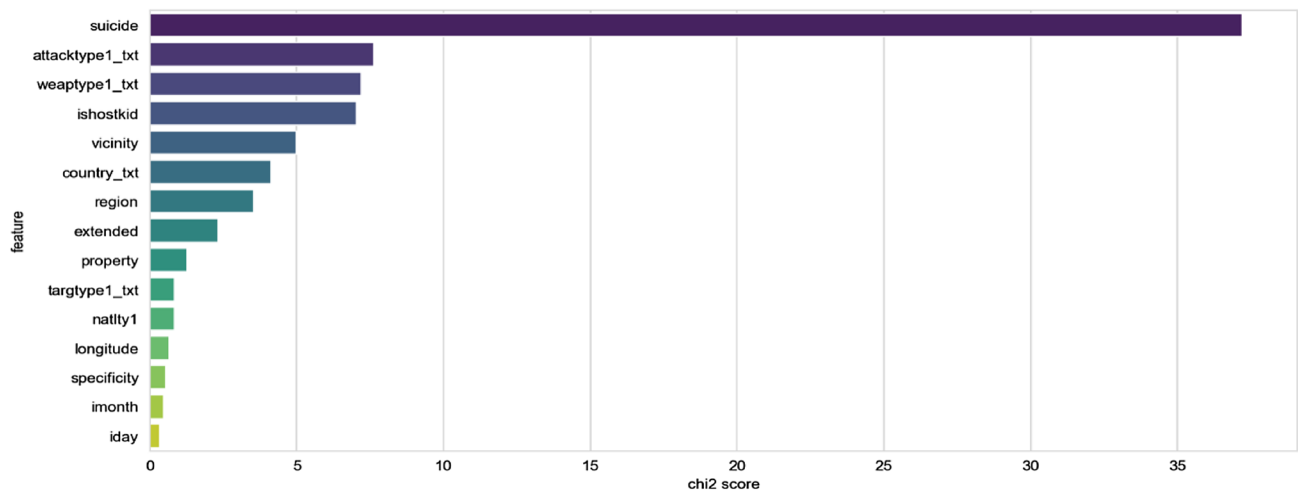**Fig. 4** Cumulative explained variance

Fig. 5 Importance feature chi-scores

ensemble, which is a serial structure. In our proposed algorithm, first, we assign the same weight $w_i (i = 1 \ldots n)$ to each sample point, and update the weight of each sample point by continuously training iterations of the training set. If the classification is wrong, the weight value of the sample point will increase, and conversely, it will decrease accordingly. In general, this makes up for the shortcomings of the previous classifier process by continuously using the current weak classifier, such that the final ensembled strong classifier achieves the target [14]. Adaboost is also a typical member of the boosting algorithm [17], and Adaboost is one of the contrast algorithms in our experiments.

Extreme gradient boosting (XGBoost) developed by Chen et al. is a machine learning system based on the improved gradient boosting decision tree (GBDT) [7]. XGBoost supports not only row sampling, but also column sampling (feature sampling), whereas GBDT only supports row sampling (sample sampling). In addition, XGBoost supports parallel computing, and allows the gradient tree to break through its limits, thus enabling fast, efficient, and greatly improved performance. In what follows, we briefly introduce the framework of the proposed algorithm GA-XGBoost, which is a part of RP-GA-XGBoost.

GA-XGBoost is a tree ensemble model composed of multiple boosting trees [29]. Since a single tree is commonly not enough to obtain good results, multiple trees can be used. Eventually, the results of each tree are summed up to obtain better results. GA-XGBoost adds the best tree model to the current classification model in the next prediction. Therefore, the prediction result can be computed as:

$$Y_i^{(k)} = Y_i^{(k-1)} + f_k(x_i), \tag{2}$$

where $f_k(x_i)$ is the best tree model in the k-th prediction, Y_i^k denotes the new prediction model, and $Y_i^{(k-1)}$ represents the current classification model. The core of the loss function is to measure the generalization ability of the model. In GA-XGBoost, model complexity is introduced to measure the efficiency of the model. Therefore, the objective function of GA-XGBoost can be written as traditional loss function plus model complexity; the objective function of GA-XGBoost can be described as:

$$Obj^k = \sum_{i=1}^{m} l\left(y_i, Y_i^{(k)}\right) + \sum_{k=1}^{K} \Omega(f_k). \tag{3}$$

The first term in Eq. (3) represents the traditional loss function, in which $l(*)$ measures the difference between the real label y_i and the predicted value Y_i. The second term in Eq. (3) denotes the complexity of the model, in which $\Omega(f_k) = \gamma T + \frac{1}{2}\lambda\|w\|^2$ is the regularization term and a penalty for model complexity, and γ denotes the penalty coefficient, T denotes the number of leaves, λ is a fixed coefficient, and $\|w\|^2$ is the L2 norm of leaf scores. The more complex the model, the larger the value of the second item will be. The second item can be good at preventing overfitting of the model. The objective function expressed by Eq. (3) is difficult to be optimized in the function space directly. Based on Eqs. (2) and (3), the objective function of GA-XGBoost at the step k can be reformulated as:

$$Obj^k = \sum_{i=1}^{m} l\left(y_i, Y_i^{k-1} + f_k(x_i)\right) + \Omega(f_k). \tag{4}$$

The goal of GA-XGBoost is to minimize Obj^k to obtain the optimal Y_i^k, which aims at minimizing the error between the prediction value and the actual value, as well as the complexity of the model. GA-XGBoost uses Taylor expansion second-order gradient statistics. Let g_i and h_i be the first and second derivatives of $Y_i^{(k-1)}$ on the loss function $l(y_i, Y_i^{(k-1)})$, respectively. By removing the constant term, a more concise target can be obtained as:

$$Obj \cong \sum_{i=1}^{m} \left[l\left(y_i, Y_i^{k-1}\right) + g_i f_k(x_i) + \frac{1}{2} h_i f_k^2(x_i) \right] + \Omega(f_k) + constant. \tag{5}$$

Fig. 6 Boosting algorithm
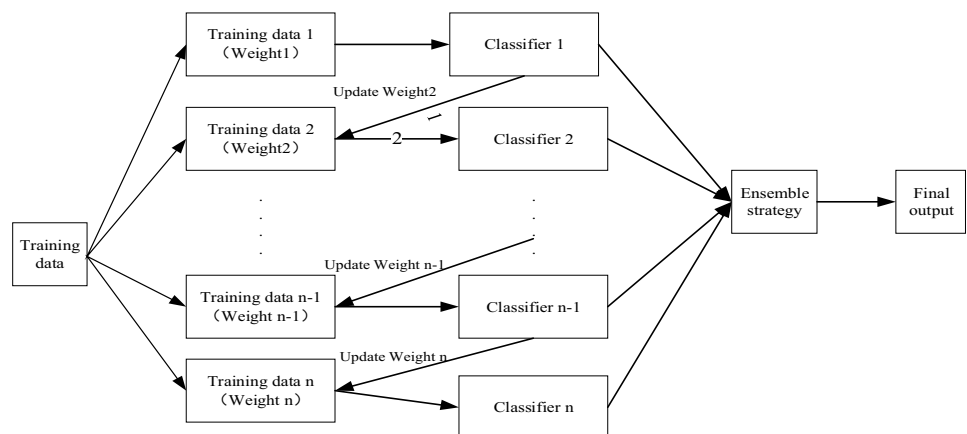
We denote by $q(x)$ the corresponding leaf node (index) to which each sample is mapped, and by $w_{q(x)}$ the score of the sample, i.e., the model prediction value. Therefore, $f_k(x_i)$ can be written as $w_{q(x)}$, $q \in \{1, 2, 3 \ldots, T\}$. The instance set of the leaf node is defined as $I_j = i|q(x_i) = j$. As a consequence, we can reformulate Eq. (5) as follows:

$$\text{Obj} \cong \sum_{i=1}^{m} \left[l(y_i, Y_i^{k-1}) + g_i f_k(x_i) + \frac{1}{2} h_i f_k^2(x_i) \right] + \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2. \tag{6}$$

In order to reflect the direct relationship between the number of leaf nodes and the objective function value, the objective function will be transformed as:

$$\text{Obj} = \sum_{j=1}^{T} \left[\left(\sum_{i \in I_j} g_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right] + \gamma T, \tag{7}$$

where I_j represent the samples on the leaf j.

Let $G_j = \sum_{i \in I_j} g_i$, $H_j = \sum_{i \in I_j} h_i$. It is clear that the optimal solution of Eq. (6) is $w_j^* = -\frac{G_j}{H_j + \lambda}$. The objective function of GA-XGBoost can be rewritten as:

$$\text{Obj} = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T. \tag{8}$$

The best tree structure f_k has been achieved. The core problem of GA-XGBoost is to determine the optimal tree structure by looking for the best branch. A simple search algorithm is summarized as follows: Enumerate all possible tree structures, calculate scores according to Eq. (8), and then find the optimal tree structure. However, the enumeration method is expensive in calculating the cost.

To determine the optimal tree structure and reduce the computational cost, GA-XGBoost uses a greedy search algorithm. The strategy of the algorithm is that GA-XGBoost starts at the root node and iteratively decomposes these features according to their ranking. This algorithm is also called "exact greedy algorithm for splitting finding" [7]. For each node on the tree, GA-XGBoost will try to add splits, then the gain after adding splits can be calculated by the following formula. That is, the change of the objective function value can be defined as follows:

$$\begin{aligned} G_{\text{obj}} &= \frac{1}{2} \left[\frac{\left(\sum_{i \in I_L} g_i \right)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{\left(\sum_{i \in I_R} g_i \right)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{\left(\sum_{i \in I} g_i \right)^2}{\sum_{i \in I} h_i + \lambda} \right] - \gamma, \\ &= \frac{1}{2} \left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{\left(G_L^2 + G_R^2 \right)^2}{H_L + H_R + \lambda} \right] - \gamma, \end{aligned} \tag{9}$$

where I_L and I_R represent the set of samples in the left and right nodes after branching, respectively, I is the union set of I_L and I_R, and the parameter γ is a complexity control index that penalizes the complexity and additional splits of tree, γ can effectively prevent overfitting. If $G_{\text{obj}} > 0$, a further branch is needed, and G_{obj} will be reduced. This process is repeated until $G_{\text{obj}} < 0$.

XGBoost contains more than 25 hyperparameters. Different hyperparameters have different functions, making hyperparameter optimization of XGBoost an extremely complicated problem. Table 2 describes some of the important hyperparameters in XGBoost. The genetic algorithm borrows the viewpoint of biogenetics, which is composed of a group of individuals called population. The chromosomes contain all the genetic information of the individual. The individual (chromosome) in the biological population reflects the strength (fitness) of the individual's survival environment ability. The current genetic information is carried to the next generation through selection, crossover and mutation, such that the low fitness solution is gradually eliminated by this continuous cycle. Genetic algorithm belongs to the adaptive probability search strategy, and its selection, crossover, mutation and other operations are performed in a

Table 2 XGBoost hyperparameters (part)

Hyperparameter	Default	Interval	Explain
tree_num	–	$[1,\infty]$	Number of weak classifiers
learning_rate	0.1	$[0,1]$	Weight of each step is adjusted to improve the robustness
max_depth	6	$[0,\infty]$	Larger the depth of the tree, the easier for the model to overfit
min_child_weight	1	$[0,\infty]$	Larger the value of min_child_weight, the model will be local optimum
Gamma	0	$[0,\infty]$	Minimum loss function degradation required for node splitting
subsample	1	$(0,1)$	Subsample ratio of the training set
colsample_bytree	1	$(0,1)$	Subsampling of columns, it can avoid overfitting
Lambda	1	–	Penalty L2
Alpha	0	–	Penalty L1
…	…	…	…

probabilistic manner, which increases the flexibility of the search process. Crossover and mutation greatly improve the search ability and diversity of genetic algorithm. Due to the excessive hyperparameters, XGBoost has the disadvantages of slow iterating and converges to a local minimum easily. Genetic algorithms that have better robustness and conciseness, compensate for these shortcomings of XGBoost, and can make the XGBoost more stable and better fitting.

In chromosome design of GA-XGBoost, we use binary strings to represent chromosomes, and XGBoost hyperparameters need to be optimized during model training. The first seven hyperparameters are selected for research and discussion. We use $P = \min_P + \left(\left| \frac{\text{sub}(\max_P, \min_p)}{2^l - 1} \right| \right) \times d$ to convert the genotype into a phenotype, where P is the phenotype of the binary string of the actual value of the hyperparameter, $\max_P$ and $\min_p$ represent the maximum and minimum values of the XGBoost hyperparameter, respectively. l denotes the length of bit string, and d is a binary string of decimal values. In terms of fitness function, this paper uses the model evaluation indicator as a fitness function. The pseudo of GA-XGBoost is shown in Table 3.

The performance of GA-XGBoost is evaluated by experimenting with different evaluation indicators as fitness functions. In the process of crossover, genetic algorithm generally has three ways of crossing. Figure 7 illustrates these ways of crossover, which greatly improve the search ability of the genetic algorithm through crossover. In this experiment, a uniform crossover is applied. The parameter values of genetic algorithm and hyperparameter intervals of XGBoost are determined according to a large number of experiments, which are described as follows. The initial population size is 20, the number of iterations is 50, the crossover probability is 0.6, and the mutation probability is 0.01. The tuning range of each hyperparameter is shown in Table 4.
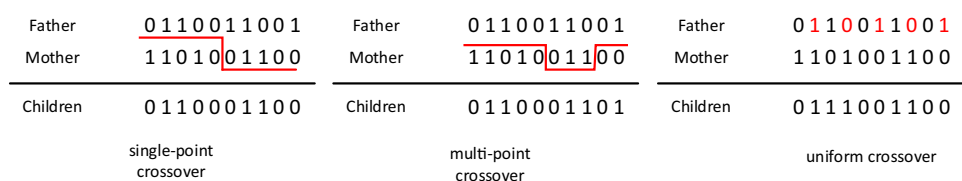
Model evaluation indicators

K-fold cross validation

Cross validation is a method of evaluating the predictive performance of a model. In actual training, the model performs well in the training set, but the result of fitting is poor in the test set. *K*-fold cross validation can reduce

Table 3 Process of the GA-XGBoost hyperparameters tuning mechanism

GA-XGBoost algorithm
Input: The training dataset D, the total number of iterations T, the size of the population G, crossover probability X, mutation probability M
Output: The best hyperparameter value set of XGBoost
1. $t \leftarrow 0$
2. Random initialize population
3. While $t < T$ do
4. $t \leftarrow t + 1$
5. For $g = 1$ to G do
6. Obtain the hyperparameter value set of XGBoost from the gth individual
7. For $l = 1$ to 10 do
8. Divide D into 10 part, where 1 as a testing set CS, 9 as a training set XS
9. Training the XGBoost on the training set XS
10. Obtain the predicted value using trained XGBoost on the testing set CS
11. Calculate the values of true positive, true negative, false positive, false negative
12. End for
13. Calculate the fitness value of the gth individual according to the fitness function
14. End for
15. Choose two individuals by roulette wheel selection
16. Conduct uniform crossover of the selected individuals with crossover probability C
17. Conduct mutation of the new individuals with mutation probability M
18. Generate the new population
19. End while
20. Return the best hyperparameter value set of XGBoost

Fig. 7 The approach of crossover

	Father	0 1 1 0 0 1 1 0 0 1	Father	0 1 1 0 0 1 1 0 0 1	Father	0 1 1 0 0 1 1 0 0 1
	Mother	1 1 0 1 0 0 1 1 0 0	Mother	1 1 0 1 0 0 1 1 0 0	Mother	1 1 0 1 0 0 1 1 0 0
	Children	0 1 1 0 0 0 1 1 0 0	Children	0 1 1 0 0 0 1 1 0 1	Children	0 1 1 1 0 0 1 1 0 0
		single-point crossover		multi-point crossover		uniform crossover

Table 4 Experimental hyperparameters tuning range	Hyperparameter name	Interval	Hyperparameter name	Interval
	The range of values for n_estimators	[10, 1000]	Range of values for subsample	[0.1, 1]
	The range of values for learning rate	[0.01 ,1]	Range of values for gammaValue	[0.1, 1]
	The range of values for max_depth	[1, 15]	Range of values for colSampleByTree	[0.1, 1]
	The range of values for minChildWeight	[1, 10]	–	–

the overfitting of the model to some extent, which is one of the model performance evaluation methods and widely used in machine learning and statistics. *K*-fold cross validation divides the training set into *k* disjoint subsets, and then takes out *k*-1 subsets to train the model each time and test the remaining subset. Finally, the mean value of the indicator values obtained by *k* operations is taken as the estimation of the method. The optimal model determined by the onefold cross validation has contingency and instability [46], and the results of fivefold cross validation and tenfold cross validation are not significantly different [9]. Tenfold cross validation is more widely used than fivefold cross validation in studies, and this paper uses tenfold cross validation to conduct the experiments in order to make the model more stable. Figure 8 shows a schematic of tenfold cross validation.

Evaluation indicators of classification

In machine learning, a confusion matrix, also known as the likelihood table or the error matrix, is commonly used to

evaluate classification performance [51]. It contains actual conditions and prediction, which reflects the classification result of the model. We use the confusion matrix to evaluate the performance of RP-GA-XGBoost, which is shown in Fig. 9. Each column of the matrix represents a prediction category, and each row represents a true category. There are four indicators in the confusion matrix:

(1) True positive (TP): The actual value is positive and the predicted value is positive.
(2) False positive (FP): The actual value is negative and the predicted value is positive.
(3) True negative (TN): The actual value is negative and the predicted value is negative.
(4) False negative (FN): The actual value is positive and the predicted value is negative.

In the experiment, "positive" means that the terrorist attack will cause casualties, and "negative" denotes otherwise. Accuracy is the correct classification ratio, which is defined as follows:
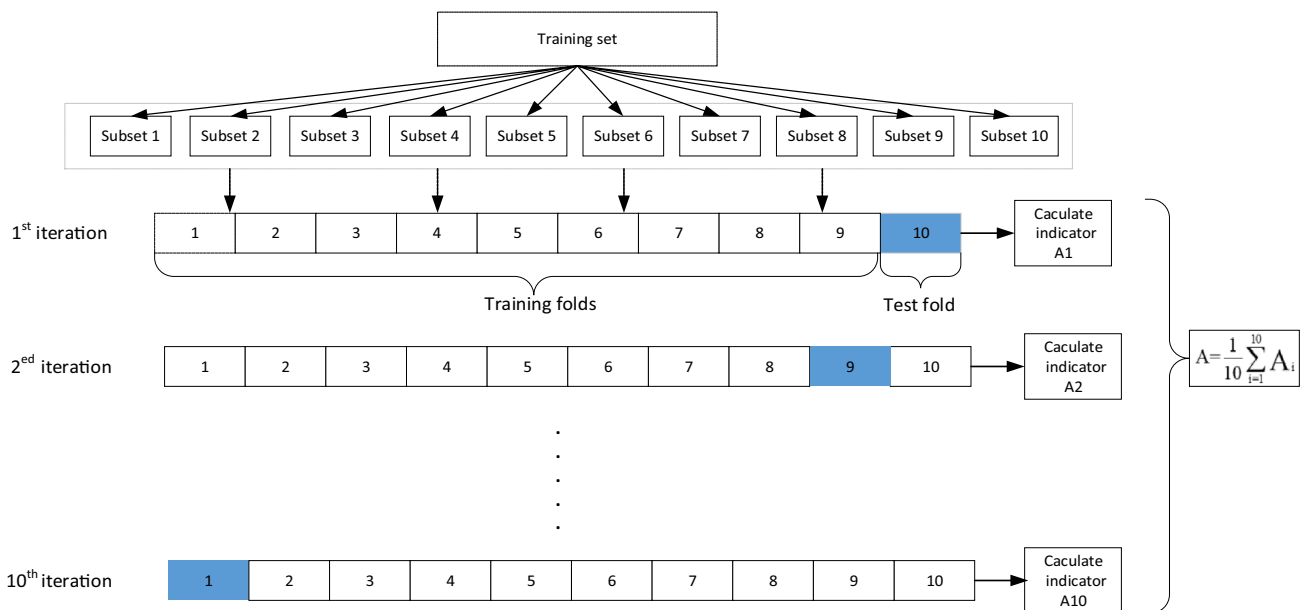

Fig. 8 Tenfold cross validation

Fig. 9 The confusion matrix

Confusion Matrix		Prediction	
		Positive	Negative
Actual Condition	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

$$\text{Accuracy} = \frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$$
$$= \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{10}$$

Sensitivity (recall) refers to the proportion of the samples that are correctly predicted in all actual "positive" samples. The "meaning" expressed in this experiment is that we can correctly predict the ability in the event of a casualty in a terrorist attack, which is described as follows:

$$\text{Sensitivity} = \text{Recall} = \frac{\sum \text{True positive}}{\sum \text{Actual positive}} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{11}$$

Specificity represents the proportion of the samples that are correctly predicted in all actual "negative" samples. Its expression can be written as:

$$\text{Specificity} = \frac{\sum \text{True negative}}{\sum \text{Actual negative}} = \frac{\text{TN}}{\text{TN} + \text{FP}}. \tag{12}$$

Precision represents the proportion of the samples that are correctly predicted in all predicted "positive" samples. Its expression can be written as:

$$\text{Precision} = \frac{\sum \text{True positive}}{\sum \text{Predicted positive}} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{13}$$

In general, the sensitivity and precision are trading off and taking turns. When the sensitivity rises, the precision will decrease, i.e., the sensitivity has to pay for the increase in the precision. F-measure comprehensively considers the sensitivity and precision [35]. Its expression can be described as:

$$F_{\beta}\text{-measure} = (1 + \beta^2)\frac{\text{Precision} \times \text{Sensitivity}}{\beta^2 \times \text{Precision} + \text{Sensitivity}}. \tag{14}$$

By adjusting the parameter β, we can change the weights of the sensitivity and precision. When $\beta = 1$, we calculate the harmonic mean of these two indicators, which is called the F1-score or the balanced F score. In addition to its average function, it gives higher scores to models with closer sensitivity and precision. If the model's gap between sensitivity and precision is too large, then it has no significant reference value.

G-mean is another indicator, that is the aggregate average of all class sensitivity rates. The higher the value of the G-mean, the better the classification performance. Its expression is as follows:

$$G\text{-mean} = \sqrt{\text{Recall} \times \text{Specificity}} = \sqrt{\frac{\text{TP}}{\text{TP} + \text{FN}} \times \frac{\text{TN}}{\text{TN} + \text{FP}}}. \tag{15}$$

Receiver operating characteristic (ROC) curve and area under curve (AUC) are commonly used to evaluate the quality of classifiers. ROC curve is shown in Fig. 10. The horizontal axis of the ROC curve are false positive rate (FPR) $\left(\text{FPR} = \frac{\text{FP}}{(\text{FP}+\text{TN})}\right)$ and the vertical coordinates are true positive rate (TPR) $\left(\text{TPR} = \frac{\text{TP}}{(\text{TP}+\text{FN})}\right)$. Each point on the ROC curve expresses FPR and TPR that correspond to a certain threshold. The closer the ROC curve is to the upper left corner, the better quality of the classification model will be. AUC is defined as the area under the ROC curve (area below the red line), which can better reflect the generalization ability of the classifier [28].
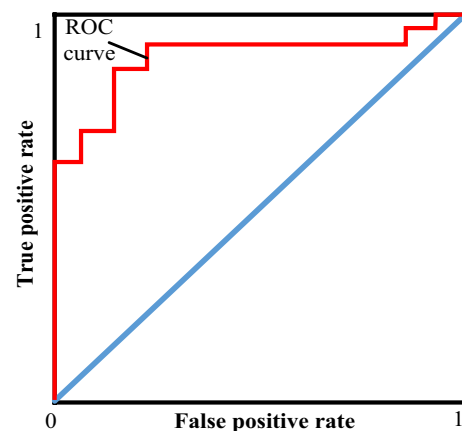

Fig. 10 Example of ROC curve

Suppose that the ROC is a curve formed by connecting points $\left\{ \left(x_1, y_1 \right), \left(x_2, y_2 \right), \dots, \left(x_m, y_m \right) \right\} \left(x_1 = 0, x_m = 1 \right)$, m^+ and m^- are positive and negative examples, respectively; M^+ and M^- represent the set of positive and negative examples, respectively. Then the calculation formula of AUC can be written as follows:

$$\text{AUC} = 1 - \frac{1}{m^+ m^-} \sum_{x^+ \in M^+} \sum_{x^- \in M^-} \left(\left(f\left(x^+ \right) < f\left(x^- \right) \right) + \frac{1}{2} \left(f\left(x^+ \right) = f\left(x^- \right) \right) \right).$$

(16)

The value of AUC ranges from 0 to 1. The larger the AUC value, the better the effect of the classifier and the stronger the generalization performance of the classifier will be.

Experimental results

Model training

After the data preprocessing, the model is trained and tested using Python software and Python third-party libraries (numpy, pandas, etc.). The experimental device is Intel CoreTM i7. Processor @2.40GHZ, window10, 16G operating system. We compare the proposed method with traditional machine learning methods, including decision tree, GBDT, logistic regression, XGBoost, random forest, Adaboost, SVM, and Naïve Bayes. In order to avoid overfitting of the model and better evaluate the generalization ability of the model, all the machine learning algorithms in this experiment use tenfold cross validation, and the whole process is repeated 20 times to avoid contingency. The experimental

results will be evaluated using the model evaluation indicators introduced in "Model evaluation indicators" (accuracy, sensitivity, precision, F1-score, AUC, G-mean).

Comparison between different feature selection methods

We conduct a large number of experiments in this subsection to demonstrate the superiority of RF-PCA, and the performance of without feature selection (WFS), Chi-square filtering and RF-PCA in different algorithms are evaluated. The experimental results are summarized in Table 5, which leading us to draw the following conclusions: (i) results of Chi-square filtering are similar to those of WFS. WFS uses 21 features to train the model; however, Chi-square filtering uses only nine features. The fewer the number of features of model training, the higher the efficiency of the model. Therefore, Chi-square filtering is superior to the WFS. (ii) By comparing the results of Chi-square filtering and RF-PCA, we can see that both methods use nine features for training; however, the evaluation indicators of RF-PCA are much higher than Chi-square filtering. This indicates that the redundant features not only reduce the training efficiency of the model, but also increase its complexity. RF-PCA eliminates the redundant features and can further improve the performance of the model by finding the optimal feature subset. (iii) The optimal values of all the evaluation indicators appear in RF-PCA, indicating that the feature selection by RF-PCA can greatly improve the performance and generalization ability of the model.

Table 5 Experimental results of different feature selection

Selection method	Number of features	Training model	Accuracy	Sensitivity	Precision	F1-score	AUC	G-mean
Without feature selection	21	Decision tree	0.767	0.686	0.792	0.741	0.777	0.747
		Random forest	0.810	0.761	0.829	0.776	0.783	0.798
		XGBoost	0.817	0.778	0.825	0.781	0.840	0.804
		Logistic regression	0.767	0.763	0.751	0.746	0.786	0.769
		GA-XGBoost	0.760	0.791	0.752	0.762	0.759	0.797
Chi-square filtering	9	Decision tree	0.737	0.696	0.716	0.703	0.761	0.706
		Random forest	0.717	0.712	0.678	0.691	0.763	0.695
		XGBoost	0.757	0.743	0.727	0.720	0.796	0.735
		Logistic regression	0.743	0.743	0.703	0.731	0.776	0.723
		GA-XGBoost	0.701	0.702	0.731	0.717	0.707	0.716
RF-PCA	9	Decision tree	0.800	0.751	0.874	0.808	0.846	0.794
		Random forest	0.840	0.797	0.867	0.831	0.839	0.825
		XGBoost	0.860	0.850	**0.860**	0.855	0.855	**0.861**
		Logistic regression	0.787	0.777	0.788	0.782	0.787	0.777
		GA-XGBoost	**0.863**	**0.872**	0.854	**0.863**	**0.870**	**0.861**

The best result in term of each indictor is given in bold

Comparison between training models with different fitness evaluations

After determining the optimal feature subset through RF-PCA, fitness functions composed of different indicators are evaluated, and the optimal fitness function is brought into RP-GA-XGBoost as a prior knowledge. Table 6 provides various formulations of the fitness functions, and Table 7 shows the performance of different fitness functions. We observe that the results of different fitness functions have some similarities. However, we will pay more attention to AUC and sensitivity among all the evaluation indicators, and the explanations are as follows.

In the terrorist attacks incidents analysis, we will pay more attention to the performance of sensitivity, since we do not hope that in the actual situations, terrorist behaviors will bring casualties to innocent people, which, however, has not been predicted. Next, the generalization ability of the model is one of our key concerns, and we hope that the model can work in other locations. When sensitivity is used as the fitness function, the sensitivity reaches 87.8% and the AUC reaches 83.4%. When AUC is used as the fitness function, the sensitivity reaches 83.1% and the AUC reaches 89.7%. However, when F1-score is used as the fitness function, the sensitivity reaches 87.2%, and it is only 0.6% lower than the highest value. The AUC

reaches 87.0%, and RP-GA-XGBoost-F1 performs well on other indicators such as accuracy. That is why we finally choose F1-score as a fitness function. It has a slight little difference from the optimal values of RP-GA-XGBoost-Sensitivity and RP-GA-XGBoost-AUC, and performs well on other evaluation indicators.

Comparison between different hyperparameter tuning methods

In this subsection, we analyze the effectiveness of the proposed method for tuning hyperparameters in comparison with the commonly used hyperparameter tuning methods, including the grid search algorithm, manual hyperparameter tuning method and Bayesian hyperparameter optimization algorithm. Table 8 lists the results of grid search algorithm, manual hyperparameter tuning method, Bayesian hyperparameter optimization in different algorithms, and results of four different tuning methods applied to XGBoost.

By comparing the results of these four methods, we draw the following conclusions. First, the results of grid search algorithm and manual hyperparameter tuning method are roughly similar. However, the labor costs of manual hyperparameter tuning method are huge, so the grid search algorithm is superior to the manual hyperparameter tuning method. Second, the results of Bayesian hyperparameter optimization are on average 0.5–1% higher than manual hyperparameter tuning method and grid search algorithm. Therefore, Bayesian hyperparameter optimization is superior to grid search algorithm. Third, among the first three methods, the optimal values mainly appear in the two algorithms of XGBoost and GBDT. Compared with traditional machine learning algorithms, XGBoost and GBDT perform better than other algorithms in accuracy and AUC. Finally, for XGBoost, we compare the results of grid search algorithm, manual hyperparameter tuning method, Bayesian hyperparameter optimization and RP-GA-XGBoost, and find that RP-GA-XGBoost in accuracy, sensitivity, F1-score, AUC is higher than other methods. This indicates that RP-GA-XGBoost is better than Bayesian hyperparameter optimization, and is a more effective method for tuning hyperparameters.

Table 6 Six types of fitness function

Fitness function name	Function
Accuracy	$f = \frac{1}{10}\left[\sum_{i=1}^{10}\left(\frac{TP+TN}{TP+TN+FP+FN}\right)_i\right]$
Sensitivity	$f = \frac{1}{10}\left[\sum_{i=1}^{10}\left(\frac{TP}{TP+FN}\right)_i\right]$
Precision	$f = \frac{1}{10}\left[\sum_{i=1}^{10}\left(\frac{TP}{TP+FP}\right)_i\right]$
F1-score	$f = \frac{1}{10}\left[\sum_{i=1}^{10}\left(\frac{2\times TP}{2\times TP+FP+FN}\right)_i\right]$
AUC	$f = \frac{1}{10}\left[\sum_{i=1}^{10} AUC_i\right]$
G-mean	$f = \frac{1}{10}\left[\sum_{i=1}^{10}\left(\sqrt{\frac{TP}{TP+FN}\times\frac{TN}{TN+FP}}\right)_i\right]$

Table 7 Experimental results of RP-GA-XGBoost with different fitness functions

Training model	Accuracy	Sensitivity	Precision	F1-score	AUC	*G*-mean
RP-GA-XGBoost-Accuracy	**0.865**	0.838	0.870	0.861	0.860	0.844
RP-GA-XGBoost-Sensitivity	0.833	**0.878**	0.802	0.822	0.834	0.853
RP-GA-XGBoost-F1	0.863	0.872	0.854	**0.863**	0.870	0.861
RP-GA-XGBoost-AUC	0.812	0.831	0.817	0.820	**0.897**	0.824
RP-GA-XGBoost-Precision	0.843	0.804	**0.873**	0.821	0.833	0.831
RP-GA-XGBoost-G-mean	0.857	0.865	0.863	0.857	0.857	**0.864**

The best result in term of each indictor is given in bold

Table 8 Results of different hyperparameter tuning methods

Hyperparameter tuning method	Training model	Accuracy	Sensitivity	Precision	F1-score	AUC	*G*-mean
Grid search algorithm	Decision tree	0.791	0.747	0.753	0.813	0.844	0.777
	Random forest	0.823	0.810	0.828	0.804	0.823	0.822
	GBDT	0.841	**0.838**	0.844	0.833	**0.843**	**0.840**
	XGBoost	**0.843**	0.837	0.838	**0.838**	**0.843**	0.839
	Logistic regression	0.726	0.764	0.706	0.737	0.727	0.724
Manual hyperparameter tuning method	Decision tree	0.785	0.770	0.812	0.791	0.833	0.805
	Random forest	0.823	0.797	0.837	0.817	0.823	0.808
	GBDT	0.839	**0.837**	0.840	0.838	0.840	0.838
	XGBoost	**0.843**	**0.837**	**0.844**	**0.840**	**0.843**	**0.840**
	Logistic regression	0.786	0.770	0.788	0.779	0.786	0.779
Bayesian hyperparameter optimization	Decision tree	0.800	0.751	0.874	0.808	0.846	0.789
	Random forest	0.840	0.797	0.867	0.831	0.839	0.825
	GBDT	0.833	0.824	**0.869**	0.846	0.850	0.817
	XGBoost	**0.860**	**0.850**	0.860	**0.855**	**0.855**	**0.861**
	Logistic regression	0.787	0.777	0.788	0.782	0.787	0.774
Contrast XGBoost	XGBoost-Grid	0.843	0.837	0.838	0.838	0.843	0.839
	XGBoost-Manual	0.843	0.837	0.844	0.840	0.843	0.840
	XGBoost-Bayesian	0.860	0.850	**0.860**	0.855	0.855	**0.861**
	RP-GA-XGBoost	**0.863**	**0.872**	0.854	**0.863**	**0.870**	**0.861**

The best result in term of each indictor is given in bold for each hyperparameter tuning method

Comparison among RP-GA-XGBoost and other classification methods

The experimental results of decision tree, random forest, GBDT, XGBoost, logistic regression, SVM, Adaboost, Naïve Bayes and RP-GA-XGBoost-F1 are summarized in Table 9. To be fair, the data preprocessing and the feature selection methods are the same for all methods. We can see that RP-GA-XGBoost-F1 is superior to other algorithms in accuracy (0.863), sensitivity (0.872), F1-score (0.863) and AUC (0.870). Note that F1-score is selected as fitness function. RP-GA-XGBoost's accuracy, sensitivity, and AUC are 0.3%, 2.2% and 1.5% higher than XGBoost, which demonstrates that the RP-GA-XGBoost has better generalization ability. Compared to XGBoost, GBDT's performance is second. Meanwhile, we also found that SVM, Adaboost and random forest perform better than decision

tree and logistic regression, where random forest and Adaboost are both ensemble learning methods, which also shows that ensemble learning methods are superior to single classifiers. In all algorithms, Naïve Bayes performs the worst, which is not good at resolving the problem of two classifications.

In order to calculate the predict value, RP-GA-XGBoost sums the scores of the leaf nodes on all CART trees. Therefore, each tree itself is not a great predictor in RP-GA-XGBoost, which is different from RF. Moreover, by summing up all the trees, RP-GA-XGBoost can perform well. By considering the importance of model interpretability, we visualized the trees built in RP-GA-XGBoost. Figure 11 shows one of all CART trees in RP-GA-XGBoost. Prediction scores can be forecasted for each leaf node, which is also called leaf weight. In order to better understand the classification principle, we explore the individual prediction. Table 10 illustrates

Table 9 Performance comparison of traditional classification methods and RP-GA-XGBoost

Training model	Accuracy	Sensitivity	Precision	F1-score	AUC	*G*-mean
Decision tree	0.800	0.751	0.874	0.808	0.846	0.789
Random forest	0.840	0.797	0.867	0.831	0.839	0.825
GBDT	0.841	0.838	0.844	0.841	0.843	0.840
XGBoost	0.860	0.850	**0.860**	0.855	0.855	**0.861**
Logistic regression	0.787	0.777	0.788	0.782	0.787	0.774
SVM	0.820	0.849	0.817	0.832	0.824	0.827
Adaboost	0.800	0.825	0.801	0.812	0.821	0.805
Naïve Bayes	0.603	0.871	0.563	0.684	0.609	0.672
RP-GA-XGBoost-F1	**0.863**	**0.872**	0.854	**0.863**	**0.870**	**0.861**

The best result in term of each indictor is given in bold
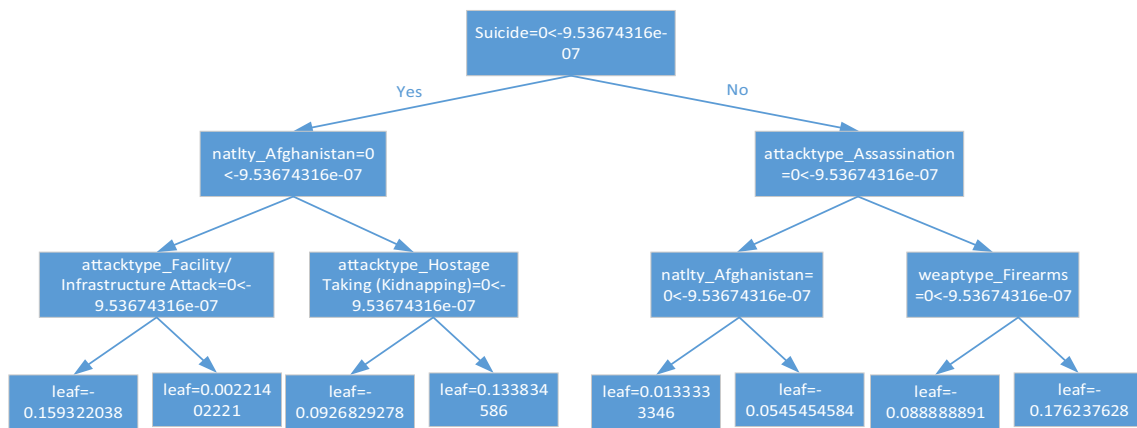
Fig. 11 One of the CARTs in the proposed algorithm

Table 10 The contribution of attributes in each class (part)

Prediction result	Weight contribution	Feature value
Death	0.300	Suicide = 1
	0.226	Attacktype_Hostage Taking (Kidnapping) = 0
	0.109	Weaptyper_Incendiary = 0
	0.077	Attacktype_Facility/Infrastructure Attack = 0
	0.044	Targtyper_Educational Institution = 0
Alive	1.110	Suicide = 0
	0.136	Targtype_Private Citizen = 0
	0.102	Targtype_Police = 0
	0.078	Natlty_Afghanistan = 0
	0.053	Attacktype_Armed Assaults = 0

the top five features that have the greatest impact on classification. Weights represent how much each feature value contributes to the final prediction of all the trees in that class.

Local terrorist attack classification results (China)

To further discuss the performance of RP-GA-XGBoost in domestic terrorist attacks, we also study terrorist attacks in China (including Taiwan and Hong Kong). The casualty prediction method introduced in "Casualty prediction method for terrorist attacks" are applied to the terrorist attacks analysis in China. The total number of terrorist attacks in China is 297, among which 188 terrorist attacks have threatened innocent people's lives. Table 11 illustrates the experimental results of RP-GA-XGBoost compared with traditional machine learning methods, and the optimal fitness function are determined according to the same experiment described in "Comparison between training models with different fitness evaluations". Compared with the global terrorist attack experiment, accuracy, sensitivity, F1-score and AUC have been improved by 8.1%, 6.8%, 9.7% and 7.2%, respectively. Furthermore, AUC achieves 94.9%, indicating again that RP-GA-XGBoost has stronger generalization ability. All these results demonstrate that RP-GA-XGBoost performs better in the domestic terrorist attack experiment, since the problem is targeted and specific. Figure 12 displays the ROC curve for each algorithm.
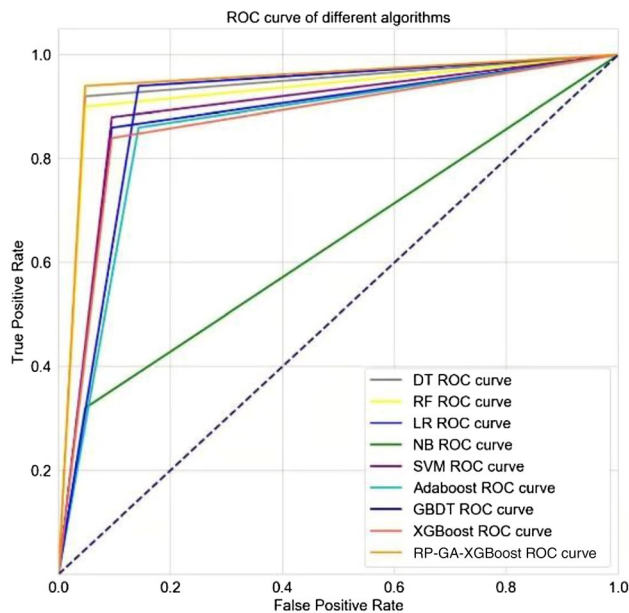
Table 11 Performance comparison of traditional classification methods and RP-GA-XGBoost

Training model	Accuracy	Sensitivity	Precision	F1-score	AUC	*G*-mean
Decision tree	0.887	0.920	0.920	0.920	0.865	0.883
Random forest	0.862	0.928	0.864	0.891	0.920	0.907
GBDT	0.915	0.940	0.940	0.940	0.898	0.910
XGBoost	0.917	0.940	0.940	0.940	0.898	0.913
Logistic regression	0.873	0.900	0.918	0.909	0.855	0.879
SVM	0.901	0.900	0.957	0.927	0.902	0.899
Adaboost	0.859	0.860	0.934	0.896	0.859	0.860
Naïve Bayes	0.507	0.320	0.941	0.478	0.636	0.543
RP-GA-XGBoost	**0.944**	**0.940**	**0.979**	**0.960**	**0.949**	**0.942**

The best result in term of each indictor is given in bold

Fig. 12 ROC curve of different algorithms

Conclusion

With the increase in terrorist attacks, governments have paid more and more attention to counter-terrorism. It is vital to quickly analyze whether or to what extent terrorist behaviors cause physical harm to innocent civilians, which can help decision makers develop emergency plans and strategies to quickly deal with terrorist attacks. To the best of our knowledge, there are few studies on terrorist casualty prediction using machine learning methods. This paper develops a XGBoost-based casualty prediction algorithm, called RP-GA-XGBoost, to predict whether terrorist attacks will cause casualties of innocent civilians. In the developed algorithm, a novel method incorporating RF and PCA is devised for selecting features, and the genetic algorithm is used to tune the hyperparameters of XGBoost. The developed method is evaluated on the public dataset and the terrorist attack dataset in China. The results demonstrate that the proposed method achieves the best performance compared with some well-known machine learning methods.

For future works, we suggest three different avenues. First, it is interesting to further verify the validity of our proposed method when the terrorist attack database has been updated. Second, the current research on terrorist attacks focuses mainly on qualitative analysis, and less on quantitative analysis. It is of interest to investigate the terrorist attacks combining qualitative and quantitative analyses. Finally, future research may also devise advanced machine learning methods such as stacking model, deep learning other than improved ensemble classifiers to predict the behavior of terrorists, and develop better security decision tools.

Appendix

See Table 12.

Table 12 List of features (part) and their description in the initial dataset

Feature name	Description
iyear	Year
imonth	Month
iday	Day
crit1	Terrorism attack criterion 1
crit2	Terrorism attack criterion 2
crit3	Terrorism attack criterion 3
latitude	Latitude of incident occurrence
longitude	Longitude of incident occurrence
extended	Whether the event duration more than 24 h
vicinity	Whether the event occurred in the immediate vicinity
doubtter	Doubt terrorism proper?
multiple	Part of multiple incident
success	Whether the incident was successful
suicide	Whether the event was a suicide attack
claimed	Claimed of responsibility?
property	Whether the attack cause property damage
propextent	Extent of property damage
propcommet	Property damage comments
ishostkid	Hostages or kidnapping victims
nkill	Total number of deaths caused by the incident
nwound	Total number of injuries caused by the incident
Specificity	Geocoding Specificity
country_txt	The country of a terrorist attack
region	The region of a terrorist attack
attcktype1_txt	The way of a terrorist attack
targtype1_txt	The target of a terrorist attack
natlty1_txt	Nationality of Target/Victim
weaptype1_txt	Type of weapon used in the event
target1	The specific attack's target
gname	Perpetrator group name
summary	A brief narrative summary of the incident
motive	The motive of incident

References

1. Adnan MN, Islam MZ (2016) Optimizing the number of trees in a decision forest to discover a sub forest with high ensemble accuracy using a genetic algorithm. Knowl Based Syst 110:86–97

2. Amin AE (2013) A novel classification model for cotton yarn quality based on trained neural network using genetic algorithm. Knowl Based Syst 39:124–132

3. Arnold J, Halpern P, Tsai M-C, Smithline H (2002) Mass casualty terrorist bombings: comparison of outcomes by bombing type. Prehospital Disaster Med 17:S47–S48

4. Basuchoudhary A, Shughart WF (2010) On ethnic conflict and origins of transnational terrorism. Def Peace Econ 21:65–87

5. Borooah VK (2009) Terrorist incidents in India, 1998–2004: a quantitative analysis of fatality rates. Terror Polit Violence 21:476–498

6. Breiman L (1996) Bagging predictors. Mach Learn 24:123–140

7. Chen T, Guestrin C (2016) XGBoost: a scalable tree boosting system. In: Proc 22nd ACM SIGKDD Int Conf Knowl Discov Data Min, KDD, vol 16, pp 785–794

8. Clauset A, Woodard R (2013) Estimating the historical and future probabilities of large terrorist events. Ann Appl Stat 7:1838–1865

9. Cui S, Wang D, Wang Y et al (2018) An improved support vector machine-based diabetic readmission prediction. Comput Methods Programs Biomed 166:123–135

10. Doğan E, Akgüngör AP (2013) Forecasting highway casualties under the effect of railway development policy in Turkey using artificial neural networks. Neural Comput Appl 22:869–877

11. Duin RPW, Tax DMJ (2000) Experiments with classifier combining rules. S16–29

12. Enders W, Sandler T, Gaibulloev K (2011) Domestic versus transnational terrorism: data, decomposition, and dynamics. J Peace Res 48:319–337

13. Fahey S, LaFree G, Dugan L, Piquero AR (2012) A situational model for distinguishing terrorist and non-terrorist aerial hijackings, 1948–2007. Justice Q 29:573–595

14. Galar M, Fernandez A, Barrenechea E et al (2012) A review on ensembles for the class imbalance problem: bagging, boosting, and hybrid-based approaches. IEEE Trans Syst Man Cybern Part C Appl Rev 42:463–484

15. Guo J, Yang L, Bie R et al (2019) An XGBoost-based physical fitness evaluation model using advanced feature selection and Bayesian hyper-parameter optimization for wearable running monitoring. Comput Netw 151:166–180

16. Hastie T, Tibshirani R, Friedman JH (2009) The elements of statistical learning: data mining, inference, and prediction, 2nd edn. Springer-Verlag, Berlin

17. Hastie T, Tibshirani R, Friedman JH (2009) The elements of statistical learning: data mining, inference, and prediction, 2nd edn. Springer-Verlag, Berlin

18. Jiang L, Kong G, Li C (2019) Wrapper framework for test-cost-sensitive feature selection. IEEE Trans Syst Man Cybern Syst. https://doi.org/10.1109/TSMC.2019.2904662

19. Khalilia M, Chakraborty S, Popescu M (2011) Predicting disease risks from highly imbalanced data using random forest. BMC Med Inform Decis Mak 11(1):51

20. Khorshid MMH, Abou-El-Enien THM, Soliman GMA (2015) Hybird classification algorithms for terrorism prediction in middle east and North Africa. Int J Emerg Trends Technol Comput Sci 4(3):23–29

21. Kis-Katos K, Liebert H, Schulze GG (2011) On the origin of domestic and international terrorism. Eur J Polit Econ 27:S17–S36

22. Kohavi R, John GH (1997) Wrappers for feature subset selection. Artif Intell 97:273–324

23. Koutanaei FN, Sajedi H, Khanbabaei M (2015) A hybrid data mining model of feature selection algorithms and ensemble learning classifiers for credit scoring. J Retail Consum Serv 27:11–23

24. LaFree G, Dugan L (2007) Introducing the global terrorism database. Terror Polit Violence 19:181–204

25. Le THM, Tran TT, Huynh LK (2018) Identification of hindered internal rotational mode for complex chemical species: a data mining approach with multivariate logistic regression model. Chemom Intell Lab Syst 172:10–16

26. Li T, Chen X, Tang H, Xu X (2018) Identification of the normal/abnormal heart sounds based on energy features and Xgboost. In: Biom Recognit, Springer International Publishing, pp S536–544

27. Lin C-T, Prasad M, Saxena A (2015) An improved polynomial neural network classifier using real-coded genetic algorithm. IEEE Trans Syst Man Cybern Syst 45:1389–1401

28. Lobo JM, Jiménez-Valverde A, Real R (2008) AUC: a misleading measure of the performance of predictive distribution models. Glob Ecol Biogeogr 17:145–151

29. Loh W-Y (2014) Fifty years of classification and regression trees. Int Stat Rev 82:329–348

30. Martha C (1981) The causes of terrorism. Comp Polit 13:379–399

31. Meng X, Nie L, Song J (2019) Big data-based prediction of terrorist attacks. Comput Electr Eng 77:120–127

32. Miksovsky P, Matousek K, Kouba Z (2002) Data pre-processing support for data mining. In: IEEE Int. Conf. Syst. Man Cybern. IEEE, Yasmine Hammamet, Tunisia

33. Mohler G (2013) Discussion of 'Estimating the historical and future probabilities of large terrorist events' by Aaron Clauset and Ryan Woodard. Ann Appl Stat 7:1866–1870

34. Nobre J, Neves RF (2019) Combining principal component analysis, discrete wavelet transform and XGBoost to trade in the financial markets. Expert Syst Appl 125:181–194

35. Pillai I, Fumera G, Roli F (2017) Designing multi-label classifiers that maximize F measures: state of the art. Pattern Recognit 61:394–404

36. Ramirez-Gallego S, Mourino-Talin H, Martinez-Rego D et al (2018) An information theory-based feature selection framework for big data under apache spark. IEEE Trans Syst Man Cybern Syst 48:1441–1453

37. Rodriguez-Galiano VF, Luque-Espinar JA, Chica-Olmo M, Mendes MP (2018) Feature selection approaches for predictive modelling of groundwater nitrate pollution: an evaluation of filters, embedded and wrapper methods. Sci Total Environ 624:661–672

38. Sachan A, Roy D (2012) TGPM: terrorist group prediction model for counter terrorism. Int J Comput Appl 44:49–52

39. Saeys Y, Inza I, Larranaga P (2007) A review of feature selection techniques in bioinformatics. Bioinformatics 23:2507–2517

40. Shafiq S, Haider Butt W, Qamar U, (2014) Attack type prediction using hybrid classifier. In: Luo X, Yu JX, Li Z (eds) Advanced data mining and applications. ADMA, (2014) Lecture notes in computer science. Springer, Cham, pp 488–498

41. Shen L, Chen H, Yu Z et al (2016) Evolving support vector machines using fruit fly optimization for medical data classification. Knowl Based Syst 96:61–75

42. Snoek J, Larochelle H, Adams RP (2012) Practical Bayesian optimization of machine learning algorithms. Adv Neural Inf Process Syst 1(2):1–13

43. Soliman OS, Tolan GM (2015) An experimental study of classification algorithms for terrorism prediction. Int J Knowl Eng 1:107–112

44. Strobl C, Boulesteix A-L, Kneib T et al (2008) Conditional variable importance for random forests. BMC Bioinform 9(1):307

45. Tao Z, Huiling L, Wenwen W, Xia Y (2019) GA-SVM based feature selection and parameter optimization in hospitalization expense modeling. Appl Soft Comput 75:323–332

46. Tibshirani R (2011) Regression shrinkage and selection via the lasso: a retrospective: regression shrinkage and selection via the lasso. J R Stat Soc Ser B Stat Methodol 73:273–282

47. Verikas A, Gelzinis A, Bacauskiene M (2011) Mining data with random forests: a survey and results of new tests. Pattern Recognit 44:330–349

48. Wang HX, Niu JX, Wu JF (2011) ANN model for the estimation of life casualties in earthquake engineering. Syst Eng Proc 1:55–60

49. Wang S, Zhu W (2018) Sparse graph embedding unsupervised feature selection. IEEE Trans Syst Man Cybern Syst 48:329–341

50. Wang Z-X (2013) A genetic algorithm-based grey method for forecasting food demand after snow disasters: an empirical study. Nat Hazards 68:675–686

51. Xia Y, Liu C, Li Y, Liu N (2017) A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring. Expert Syst Appl 78:225–241

52. Xing H, Zhonglin Z, Shaoyu W (2015) The prediction model of earthquake casualty based on robust wavelet v-SVM. Nat Hazards 77:717–732

53. Xu X, Qi Y, Hua Z (2010) Forecasting demand of commodities after natural disasters. Expert Syst Appl 37:4313–4317

54. Harari YN (2016) Homo Deus: a brief history of tomorrow. Harvill Secker, New York

55. Zhang C, Ma Y (2012) Ensemble machine learning. Springer, Boston

56. Zhang D, Qian L, Mao B et al (2018) A data-driven design for fault detection of wind turbines using random forests and XGboost. IEEE Access 6:21020–21031

57. Zhang L, Qiao F, Wang J, Zhai X (2019) Equipment health assessment based on improved incremental support vector data description. IEEE Trans Syst Man Cybern Syst 99:1–5

58. Zhang X, Mahadevan S (2019) Ensemble machine learning models for aviation incident risk prediction. Decis Support Syst 116:48–63

59. Zheng L, Shen C, Tang L et al (2013) Data mining meets the needs of disaster information management. IEEE Trans Hum Mach Syst 43:451–464