Machine Learning Engineer Nanodegree

Capstone Proposal

Chang Chia-Hua
Nctueric@gmail.com

Proposal

I. Domain Background

Fraud risk is everywhere, but for companies that advertise online, click fraud can happen at an overwhelming volume, resulting in misleading click data and wasted money. Ad channels can drive up costs by simply clicking on the ad at a large scale. With over 1 billion smart mobile devices in active use every month, China is the largest mobile market in the world and therefore suffers from huge volumes of fraudulent traffic. TalkingData, China's largest independent big data service platform covers over the 70% of active mobile devices nationwide. They handle 3 billion clicks per day, of which 90% are potentially fraudulent. Their current approach to prevent click fraud for app developers is to measure the journey of a user's click across their portfolio, and flag IP addresses who produce many clicks, but never end up installing apps. With this information, they have built an IP blacklist and device blacklist.

Fraud behavior detection is endless competition between the developer in the online advertise company and developer in some automation code tool. As a code developer, we are truly like to develop some tools to replace human maturations on computer or other mobile devices; it will be helpful to improve productivity for many companies. However, the tool will also use to something not for productivity but for profits, such as fraud click. Most of them are clicking by some computers instead of human; it just develops an algorithm to learn to click like human behavior. Therefore, it is interesting to me to study how to build a model to predict fraud click even I believe some person may be study how to cheat the model.

II. Problem Statement

The problem is quite clear: how to predict willing of the users download the apps to install after they click the advertisement. In other words, what we need is to develop a model to separate the clicks into the meaningful clicks and fraud clicks. Currently, the talking has some methods to predict the fraud click, but it still needs to improve. When we solve this problem, it could provide a better method to improve their prediction accuracy and they will always be one-step ahead of fraudsters.

III. Datasets and Inputs

For this competition, your objective is to predict whether a user will download an app after clicking a mobile app advertisement. To support your modeling, they have provided a generous dataset covering approximately 200 million clicks over 4 days! These data are downloaded on the website: https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection

The dataset file descriptions:

- train.csv - the training set
- train_sample.csv - 100,000 randomly-selected rows of training data

These data has eight features in each row of training data and training sample data, and these features are listed.

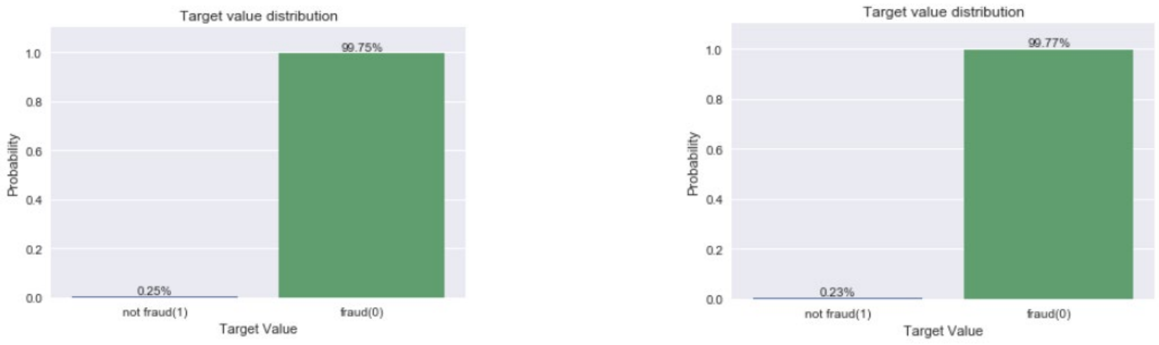
Training features:

- ip: ip address of click.
- app: app id for marketing.
- device: device type id of user mobile phone.
- os: os version id of user mobile phone.
- channel: channel id of mobile ad publisher.
- click_time: timestamp of click (UTC).
- attributed_time: if user download the app for after clicking an ad, this is the time of the app download

Target features:

- is_attributed: the target that is to be predicted, indicating the app was downloaded

These data have seven training features that are used to predict the click is a fraud or not, and one target feature 'is_attributed' that indicated the app was downloaded. The training data has total 184903890 rows, and the sample data has 100000 rows. By calculating the target distribution form these two dataset, both of these data has the same probability distribution about the fraud as shown in the fig.(a) and fig.(b). We can find the fraud probability is extremely high to 99.75%. Therefore, it may need re-sampling process to balance these data if our training model is bad to predict the results. Because these training data are extremely large for our personal computer to training, we will use the sampling data to train and divided into three group training, validation, testing. After our training process, we will use the test data download from kaggle to do the final evaluation to verify our model performance.



(a) (b)

Fig.1 The target value distribution in the whole training data(a) and the sample data(b).

IV. Solution Statement

Our goal is to develop an algorithm/model which and precisely predict whether a user will download an app after clicking a mobile app ad based on the recorded properties of that user. These is a true or false question. The performance will be evaluated on area under the Receiver operation characteristic (ROC) curve between the predicted probability and the observed target. The quality of our model is depending on how small difference between our predictions and truths. After that, the model could be used to distinguish between meaningful clicks and fraud clicks and reduced the amount of wasted money caused by fraudulence.

V. Benchmark Model

This project is actually taken from one of Kaggle competitions. They provide a benchmark model that is developed by a random forest method. The score of this benchmark model is 0.91. In addition, the score is evaluated on area under the Receiver operating characteristic (ROC) curve between the predicted probability and the observed target.

VI. Evaluation Metrics

Evaluation metric will be the area under the Receiver operating characteristic (ROC) curve between the predicted probability and the observed target. Such metric is also called as "AUC". AUC as a further interpretation of ROC is a very straightforward and easy understanding metric of a binary classifier system. Since now we are trying to establish a model to predict whether a user will download an app after clicking a mobile app or not. This is exactly a binary classification problem.

Given a threshold parameter T, the instance is classified as “positive” if X > T, and “negative” otherwise. X follows a probability density $f_1(x)$ if the instance actually belongs to class “positive”, and $f_0(x)$ if otherwise. Therefore, the true positive rate is given by $\mathrm{TPR(T)} = \int_T^\infty f_1(x)d(x)$ and the false positive rate is given by $\mathrm{FPR(T)} = \int_T^\infty f_0(x)d(x)$. The ROC curve plots parametrically TPR(T) versus FPR(T) with T as the varying parameter. Then the AUC is simply the area under the ROC. Generally, we can judge our model through the value of AUC like follows:

- AUC = 0.5 (no discrimination)
- 0.7 ≦ AUC ≦ 0.8 (acceptable discrimination)
- 0.8 ≦ AUC ≦ 0.9 (excellent discrimination)
- 0.9 ≦ AUC ≦1.0 (outstanding discrimination)

Ref: ROC curve https://en.wikipedia.org/wiki/Receiver_operating_characteristic

VII. Project Design

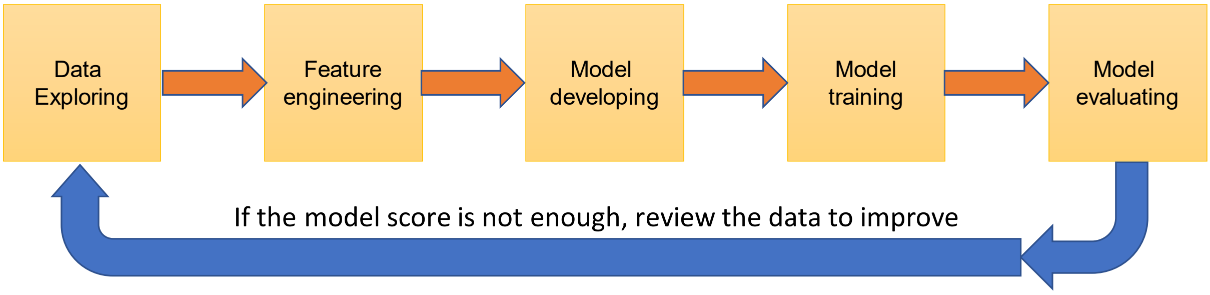


Fig.2 The model developing workflow.

In this project, my workflow will follow a traditional model developing as the fig. 2 shown. The first priority is investigating the data and developing some basic ideas about the interrelationship between some different features or native properties of each feature. Base on the exploring results, we can create some new features based on the existing features. As the next step, we need to check and clean our data. Because sometimes our data will contain some missing data or repeating data. Therefore, in this data preprocess steps for our model developing, we will need to huddle data about the missing value, outlier value, normalize for numeric features, or do some pca processing to transport feature value. If we have an categorical feature or text format feature, more data preprocessing techniques will be considered.

Before starting to develop our model, we will need to split our training data into three groups: training, validation, and testing. This step is for cross validation. The next important thing is developing a proper model for our model. Currently, we have already learned methods from the projects of the ‘boston housing’ and ‘finding donors’, included the supervised and unsupervised learning model. In these cases, it may look like a supervised learning problem.

I will try to focus on building two kinds of classifiers: one is based on the neural network, and the other is based on the random forest. For the neural network, I will construct 2- 3 fully connected layers and try different activation functions for hidden layers. Then the output layer will pass through a sigmoid function for converting the output values as probabilities. Other parameters like optimizer and loss function(s) are to be tuning.

For the random forest, it is an ensemble learning method, which operate by constructing a multitude of decision trees as collecting the contributions from many weak learners. It keeps the advantages of decision tree and makes the final mode more general. That is, with proper parameter setting of each decision trees, random forest can effectively avoid overfitting the training data. Grid search method will be optional for finding the best combination of model's parameters.

Once we have finished training our models, we will use the evaluation metric, AUC, to evaluate our training models. The models will be review thoroughly to check if anything need to improving, and the Kaggle's officical evaluation method will be taken into account. Once our model is get the acceptable score is at least over 0.91 (outstanding discrimination, we can publish that model.

Ref: https://en.wikipedia.org/wiki/Artificial_neural_network
Ref: https://en.wikipedia.org/wiki/Random_forest