TalkingData AdTracking Fraud Detection

DS502 Project Proposal

March 20, 2018

Eduardo Calle (ecalleo@gmail.com), Nikolaos Kalampalikis (nkalampalikis@wpi.edu), Anastasia Leshchyk (aleshchyk@wpi.edu), Ahmad "Daniel" Moghimi (amoghimi@wpi.edu), Michael Moukarzel (mamoukar@wpi.edu)

Problem Statement

Pay-per-click (PPC) is a known internet advertising model, in which an advertiser pays a publisher to direct traffic to websites or mobile applications. In this model, fraud risk is inevitable if the advertiser can not distinguish legitimates clicks from fraudulent ones. 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 on 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 70% of active mobile devices nationwide. They handle 3 billion clicks per day, of which 90% are potentially fraudulent. Their current approach to preventing click fraud for app developers is to measure the journey of a user's click across their portfolio, and flag IP addresses who produce lots of clicks, but never end up installing apps. With this information, they've built an IP blacklist and device blacklist.

While partially successful, TalkingData wants to always be one step ahead of fraudsters and have turned to the Kaggle community for help in further development of their click fraud detection. In their 2nd competition with Kaggle, we are challenged to build an algorithm that predicts whether a user will download an app after clicking a mobile app advertisement. To support our modeling, they have provided a generous dataset covering approximately 200 million clicks over 4 days!

Reference: TalkingData AdTracking Fraud Detection Challenge
https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection#description

Dataset

Our objective is to predict whether a user will download an application after clicking a mobile advertisement. This is a **classification** problem that outputs two classes. Our training data and test data consists of 184 million and 1.8 million click records respectively. Each row of training data contains a click record, with the following features:

1. *ip:* ip address of click.
2. *app:* app id for marketing.
3. *device:* device type id of user mobile phone (e.g., iphone 6 plus, huawei mate 7, etc.)
4. *os:* os version id of user mobile phone
5. *channel:* channel id of mobile ad publisher
6. *click_time:* timestamp of click (UTC)
7. *attributed_time:* if user download the app for after clicking an ad, this is the time of the app download
8. *is_attributed:* the target that is to be predicted, indicating the app was downloaded

Note that ip, app, device, os, and channel are encoded. The test data is similar, with the following differences:

- *click_id:* reference for making predictions
- *is_attributed:* not included

Sample Dataset:

ip	**app**	**device**	**os**	**channel**	**click_time**	**attributed_time**	**is_attributed**
87540	12	1	13	497	2017-11-07 09:30:38	NA	0
177975	45	1	4	419	2017-11-07 10:14:19	017-11-07 10:14:30	1

Methodology

1. Data analysis.
   We will plot each predictor individually and find correlations between all of them. We will use Principal Component Analysis in order to identify the important predictors and to remove the redundant ones.

2. Model exploration
   We will start with KNN, since we have high-density data. For this, we will normalize the data across predictors. Then, we will try Logistic Regression with L2 regularization, which should lead to less overfitting, and without it just to compare the results. Also, we will try Lasso, because it performs the variable selection, which gives generally easier models to interpret. Support Vector Machine is for the binary classification problems and should work great with our dataset. We will do SVM with L2 regularization. Additionally, one hidden layer neural network with non-linear activation function will be tried. L2 regularization will be performed on the neural network weights to limit their magnitude.

3. Model optimization
   We will use a hierarchical cross-validation and different error metrics in order to find the optimal coefficients.

Error Metrics

MSE, confusion matrix, ROC (receiver operating characteristic).

Challenges

Overfitting, oscillation optimizer around local minima. This is caused due to a high value of learning rate. It can be avoided by using low learning rate, but if it's reduced to a very low number, it can cause slow conversion. The challenge could be to find the optimum learning rate.