Machine Learning Engineer Nanodegree

Capstone Proposal

Danko Komlen

October 10th, 2018

Proposal

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.

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 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 lots of clicks, but never end up installing apps. With this information, they've built an IP blacklist and device blacklist. While successful, they want to always be one step ahead of fraudsters and are searching for solution via Kaggle community to help in further developing their solution.

My interest in the particular problem is related to predictions on time series data. This is something close to my daily job and I'm interested to learn more about.

Problem Statement

In the AdTracking Fraud Detection challenge[1], the goal is to build an algorithm that predicts whether a user will download an app after clicking on a mobile app ad. For every click on ad, TalkingData collects additional metadata information about the user,

[1] Kaggle competition: https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection

like IP and device. Using this data we want to build a system that will tell us the probability that the user will download the advertised app after looking through the ad. TalkingData would then be able to use such a system to detect click fraud and pay advertisers only for legitimate clicks.

Datasets and Inputs

The data used will be the one provided by the TalkingData as part of their challenge[2]. It contains click records for the ads they provide.

Dataset	**Comment**	**Size (clicks/size)**
train.csv	the training set	57.537.506 / 2.5 GB
test.csv	the test set	18.790.469 / 824 MB
test_supplement.csv	larger test set, subset of it is used to evaluate model on Kaggle	57.537.505 / 2.5GB

Every click record has following associated fields:

- click_time: timestamp of click (UTC)
- ip: ip address of click
- device: device type of user mobile phone
- os: os version id of user mobile phone
- channel: channel id of mobile ad publisher
- app: app id for marketing

Training data contains two additional fields:

- is_attributed: was app downloaded, to be predicted
- attributed_time: time of app download

For the given click data, the goal is to output the probability that the user will download the app.

[2] Datasets on Kaggle: https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection/data

Solution Statement

First we will investigate given datasets to get the senses of which fields are important for training. We will also look at distributions of every field and number of positive vs negative samples.

Additional features that we will consider adding:

- for each attribute number of clicks in previous N minutes with same attribute value
- for each attribute time since last click with same attribute value

After data investigation and feature engineering we will prepare the training and validation sets. We can categorize the problem as probabilistic binary classification using batch learning. So, the solution to the problem will be the model that will have best performance out of following candidates: logistic regression, neural network, naive bayes, tree based models. We will also investigate improving performance with ensemble methods like bagging and boosting.

Benchmark Model

Constant model that predicts no app download scores 0.5 on Kaggle evaluator. Random model that uses uniform distribution scores 0.4995 as private score (82% of test data) and 0.5003 as public score (18% of the test data). One approach to this problem is done by Elior Tal and his solution scores 0.825 on a subset of the test set[3]. In his work he compares performance of random forest and boosting algorithms. We will use this as a benchmark model as it is well documented and scores better than our initial baseline models. Another evaluation of the approach will be done against existing Kaggle solutions on the leaderboard[4].

Evaluation Metrics

For the evaluation metric we will use area under ROC curve (AUC). This metric is suitable because the data is unbalanced in favour of app not downloaded class. AUC represents the probability that a random positive example has higher prediction than a random negative example.[5] AUC ranges in value from 0 to 1.

---

[3] Benchmark model: https://rpubs.com/el16/410747
[4] Kaggle leaderboard: https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection/leaderboard
[5] AUC: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

Project Design

Project will be done in several parts with a success criteria associated to each. First part will be data investigation where we will try to answer questions like: what are the distributions of every field across train set, what fields are important for doing predictions and what features can we engineer from existing features. In second part we will prepare training and validation data from preprocessed train dataset with new features.

Next part will be researching existing work in the area and training different ML models based on the knowledge acquired. Some candidates for algorithms are: logistic regression, neural network, naive bayes, tree based models. We will also investigate improving performance with ensemble methods like bagging and boosting.

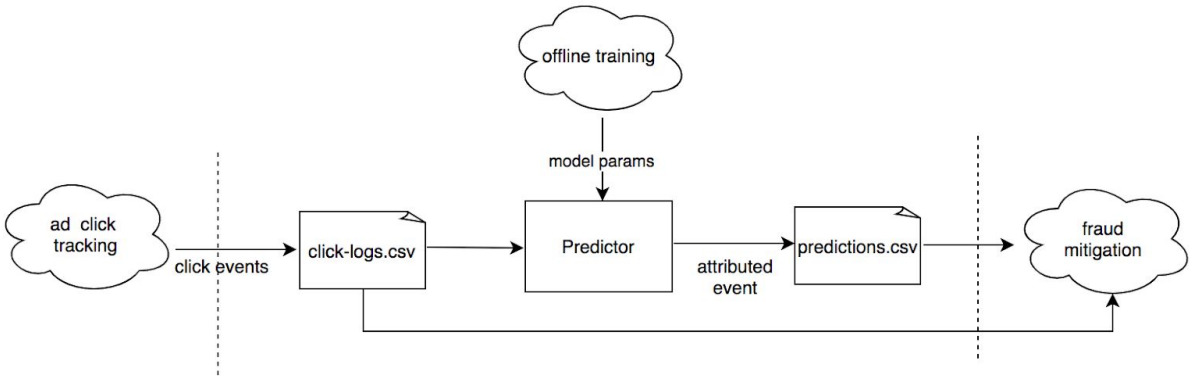

Image 1: proposed architecture

Finally the best model will be picked to be used in the predictor system for the production environment of the TalkingData. Image 1 shows the proposed final architecture that uses the trained predictor to process output of ad clicking TalkingData systems and outputs the predictions. Fraud mitigation systems can then use both click logs and predictions to prevent fraudulent clicks in the future and adjust payments to ads that were target.