Machine Learning Engineer Nanodegree

Capstone Proposal

Keyun Shang

March 11th, 2018

TalkingData AdTracking Fraud Detection Challenge (from Kaggle[1] competition)

Domain Background

For companies that advertise online, an overwhelming volume of click fraud can lead to misleading click data and wasted money. Although companies keep trying to find the behaviour pattern of click fraud , the huge amount data makes it almost impossible to be analyzed manually.

In recent years, the emerge of high computation power and machine learning has given us the ability to use computer for predicting the user behavior. [2] [3]

Problem Statement

TalkingData is china's biggest independent data service platform and had suffered from huge volumes of fraudulent traffic. Through building an IP blacklist and device blacklist , they have successfully prevented click fraud for app developers. However, now they want to step ahead of fraudsters and hope to find an user behaviour pattern that predicts if a user will download an app after clicking a mobile app advertisement.

Their problem is a binary classification problem. The input are a set of features including ip address, app, device, os ,channel, click time etc., the goal is to predict whether a user download an app after clicking a mobile app advertisement.

Datasets and Inputs

TalkingData has provided a dataset covering approximately 200 million clicks over 4 days .[4] In the 1,000 randomly-selected rows of training data, there is only 2 clicks that end up with

---

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

[2] https://pdfs.semanticscholar.org/1518/6bd13f3f5a33598be9930d2e093055bb7c93.pdf (A Novel Ensemble Learning-based Approach for Click Fraud Detection in Mobile Advertising)

[3] http://ink.library.smu.edu.sg/cgi/viewcontent.cgi?article=2989&context=sis_research (Detecting Click Fraud in Online Advertising: A Data Mining Approach)

[4] https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection/data

downloading app. From this , we can infer that about 0.2% of the clicks could lead to app download.

Input data fields

Each row of the training data contains a click record, with the following features.

- ip: ip address of click.
- app: app id for marketing.
- device: device type id of user mobile phone (e.g., iphone 6 plus, iphone 7, huawei mate 7, etc.)
- 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
- is_attributed: the target that is to be predicted, indicating the app was downloaded

We will split the training data (train.csv) into training/testing sets with a ratio of 80:20. For model tuning, we will split the training set again into training/validation set with a ratio of 80:20.

Solution Statement

First, we need to understand the relationship between the features as input variables and the target labels as output variables. Second, we will build a benchmark model as the start point for evaluation. Third, we will build and train several new models with several commonly used algorithms such as decision tree, Gassic NB, SVM etc. We will evaluate and choose the best model from them. Finally, We will optimize the best model with grid search.

Benchmark Model

As there is no available benchmark model for this project at present, we will use the naive predictor as the benchmark model. Naive predictor is a simplest predictor that predicts the data with current values. Any new model should be better than or at least the same with it.

Evaluation Metrics

As this is a Kaggle competition project, the best way for evaluation would be the Kaggle score for the test set. However, it is not appropriate in this study due to the following reasons:

1) The number of submissions per day is limited by Kaggle.
2) If the Kaggle project expires before we complete the study, we can no longer obtain Kaggle score.

For academic purpose, we will use F-beta score as the metric instead.

$$F_{\beta} = (1 + \beta^2) \cdot ((precision \cdot recall) \div (\beta^2 \cdot precision + recall))$$

Project Design

The project will consist of the following parts.

Data processing

Since the input fields such as device, app, channel are non-numeric features, we will convert them to numeric value by using the one-hot encoding scheme.

we also will split the data (both features and their labels) into training and test sets. 80% of the data will be used for training and 20% for testing.

Evaluating model performance

We will evaluate the performance of naive predictor which is the benchmark model in this project.

We will train the model with several different algorithms to compare. Since this is a classification problem, I plan to try three or four from the list below.

- Decision tree
- Gaussian Naive Bayes
- Support Vector Machines
- Ensemble Methods (Bagging, AdaBoost, Random Forest, Gradient Boosting)
- Stochastic Gradient Descent Classifier
- K-Nearest Neighbors
- Logistic Regression

Improving the performance

From the models we've tried, we will choose a model with the best performance as the target to tune.

For model tuning, We will use grid search with at least one important parameter tuned with at least three different values.

Furthermore, we also will observe the features relevance and try to extract the feature importance. Based on the observation and extraction result, we will try to reduce the features space and optimize the model with simplified information.