Machine Learning Engineer Nanodegree

Capstone Proposal on TalkingData AdTracking Fraud Detection Challenge

Sakshi
September 10, 2019

Proposal

Domain Background

Fraud risk is everywhere however, with online advertisement, adfraud can happen at an overwhelming volume resulting in misleading click data and wasted money. Online advertising is a growing, multi-billion-dollar market. It is predicted that global digital ad expenditure will reach US$225 billion — accounting for 44% of total ad expenditure. The sheer size of this market tempts criminals and hackers into creating technology and techniques to steal money from the advertisers. It is also estimated that in 2019 ad fraudsters will steal $5.8 billion from advertisers, however, because some types of ad fraud are very hard to detect, and the technology to protect advertisers is immature, the actual figure may be much higher.

Reference : https://clearcode.cc/blog/rtb-online-advertising-fraud/

Problem Statement

Talking Data is one of the most prominent big data platforms, covering more than 70% of mobile devices' activities across China. It is a third-party platform which provides analytics for small and medium-sized mobile applications. They handle 3 billion clicks per day, of which 90% are potentially fraudulent. Their strategy is to measure the journey from click to install and flag the IP's/devices that generate fake clicks which never lead to installing the app. This helped them to build a portfolio for all the fraudulent IP's and devices to be considered for blacklisting. Some mobile applications us Pay-per-click (PPC) model which help direct traffic to their applications. The sole aim is aimed at having more clicks that will help to have more installation of their applications. By having millions of clicks which are actually fraudulent will lead the advertisers to pay a hefty amount of money as they are charged based on each click. The other reason fraudster does click spamming is that by bombarding the system with millions of clicks, malicious parties tries to affect the daily operation of mobile applications as it increases the load on the server.

In the AdTracking Fraud Challenge of Kaggle, they want to determine the probability of app download by the user after clicking the ad. Low probability means it is a click fraud and high probability means a genuine user. This model might be helpful to increase their solution's accuracy in identifying fraudsters.

Datasets and Inputs

The dataset consists of around 200 million records with the following features:

Feature	Description	Feature Use
Ip	ip address of click	Input
App	app id for marketing	Input
Device	device type id of user mobile phone (e.g., iphone 6 plus, iphone 7, huawei mate 7, etc.)	Input
OS	os version id of user mobile phone	Input
Channel	channel id of mobile ad publisher	Input
Click_time	timestamp of click (UTC)	Input
Attributed_time	If installed, Install time of the app	Input
Is_attributed	the target that is to be predicted, indicating the app was downloaded	Target Variable

The data set is pre-split into training and test set. The test set has the similar features with the inclusion of an extra one named: click_id. This act as a reference for making predictions.

Solution Statement

In order to attain the final goal of finding the probability of installing the app by a click id, I would follow the following steps:

- Data Pre-processing: It is said that every data scientist should spend 80% time for pre-processing and 20% time for actual training/testing. I will also follow the same strategy. Pre-processing involves data cleaning, looking for missing data or inconsistent data. This step is called data exploration. As the data for this project is huge, so I will take a sample of data may be 100,000 records to understand it. Feature engineering is the next step once we have analysed the data properly. Machine learning algorithm will understand only the numeric

data, so converting the categorical data/ timestamp to numerical features for example: in this case, the click time can be converted to specify day of the week (weekday/weekend) which will help to generate new features like frequency of the clicks with respect to the day of the week, trend of the clicks in specific hours of the day.

- Model that best fits the type of the data. As this is a classification task, so we will look for different classifiers depending upon the number of features and samples to train. I am inclined towards Light GBM (Gradient Boosting), a fast, distributed, high-performance gradient boosting framework based on decision tree algorithm. But I would also like to try other classification algorithms available.

Benchmark Model

In machine learning, there is something known as "No-free Lunch" theorem which means no single algorithm works well for every problem. However typically for classification algorithms, Gradient Boosting machines (GBM) are proven successful across many domains and one of the leading methods for winning Kaggle competitions. Light GBMs are also popular to train large dataset as they are comparatively faster and have lower memory usage. We will try to compare different boosting algorithms with Light GBM.

Evaluation Metrics

Some common evaluation metrics for classification algorithms are:

1. True positive rate/ Recall
2. False positive rate
3. Precision
4. ROC

Project Design

The workflow is as follows:

- Exploring the data: Establish basic statistics and understanding of the dataset
- Data pre-processing and feature engineering: Perform basic cleaning and processing if needed. Identify features and design new features based on the non-numeric ones. Normalizing the data.
- Training and Model evaluation
- Model Tuning to improve and compare the result with other models.
- Conclusion

Prospects

The sheer size of the given data (over 180 million rows of data, ~7.3GB) will no doubt pose a challenge to both the data processing and training steps. So, I will try to stick to a smaller dataset (say 1/3rd) to speed up my performance and project submission.