Click Through Rate Prediction

Aditya Bhise
Master of Computer Science
North Carolina State University
Raleigh, USA
avbhise@ncsu.edu

Ankit Bhandari
Master of Computer Science
North Carolina State University
Raleigh, USA
abhanda3@ncsu.edu

Shubham Bhawsinka
Master of Computer Science
North Carolina State University
Raleigh, USA
sbhawsi@ncsu.edu

Yatish Mehta
Master of Computer Science
North Carolina State University
Raleigh, USA
yjmehta@ncsu.edu

***Abstract*— This study involves the prediction of ad click-through rate which is the most important business component of displaying advertisements on a website. The goal is to determine the most accurate machine learning algorithm for the problem. We have implemented two machine learning algorithms i.e. Random Forest and Logistic Regression for this study and predicted probability of the advertisement being clicked given some properties of ad, page and user.**

Keywords- CTR; Random Forest; Logistic Regression; Advertising;

I. INTRODUCTION

Display advertising is one the biggest businesses in the internet space. Top internet companies like Google, Facebook, and Bing depend on ads for the majority portion of their revenue. For every venture ranging from content management, B2C platforms and mobile games, ad monetization is the main source of revenue. Since the inception of internet, display ads have been popular. Flash ads, banner ads, animated games, such formats were used to display marketing content but this trend is changing. Traditionally advertisers would engage in a long term static contract with the content provider without considering the audience. But now rather than an aimless broadcast advertising campaign, advertisers want granular target audience selection. This increases the flow of converting a cold lead into a potential customer. A typical active internet user is shown more than 1700 display ads per month, but less than 2% of them are clicked. With such a low hit rate, all efforts are getting directed to get maximum user engagement.

To evaluate the effectiveness of display ads for a marketing campaign a term CTR (Click through Rate) was introduced. CTR is defined as the ratio of no. of clicks to the number of impressions made by that particular ad. If the purpose of the ad is to just get message out, advertisers pay per impression, also known as Cost per Impression (CPM). For the purpose of lead generation, advertisers and the publishers engage in a pay per click model, Cost per click (CPC), in which the advertisers pay only if the advertisement is clicked by the user. As mentioned before, with such a low hit rate the task of CTR prediction, i.e. predicting the probability of the user clicking on a particular ad becomes important. CTR prediction becomes very important for all businesses in the consumer space as it affects user experience, stake holder’s revenue and the profitability of the investing advertiser.

In this project we have taken the dataset provided by CriteoLabs. CriteoLabs works in the domain of optimizing digital advertising performance. It analyses the effectiveness of digital marketing, thus making ROI transparent. CriteoLabs has provided one week of advertising transaction data to develop a model for CTR prediction. User details have been hashed for the sake of anonymity.

II. APPROACH

To solve this particular CTR problem we have considered several algorithms and finally decided Random Forests and Logistic Regression are best suited for this particular problem over algorithms like bagging and boosting. To move ahead for actual experimentation and deciding upon which algorithm to choose we made several sample training dataset of different sizes. Then we did Exploratory Data Analysis for each attribute and due to high dimensionality of data we short listed these two algorithms. We ran both Random Forests and Logistic Regression on them and found that Random Forests gave relatively better accuracy. Later, out of many available Loss functions we choose Log Loss function for our analysis as we have a-priori knowledge that we have only two class labels and Log loss gives a good probabilistic distribution ranging in [0,1] when we have only two classes for reference.

We have loaded the data in MongoDB using ruby. Ruby was also used for basic data preprocessing. We started with weka for learning Random Forest and Logistic Regression. But we found the implementations to be very memory inefficient given the high dimensionality of data. We switched to python packages sklearn and numpy to train, validate and test the models.

III. METHODS

A. *Classifiers : Random Forest*

Random Forest is a very effective ensemble algorithm used for both classification and regression. Data is sampled with replacement similar to Bagging, that is, n records are selected with replacement from the training dataset. However, the predicate selection method is different than Bagging. In each decision tree that Bagging produces, next predictor that gives the best split is chosen from the pool of all available predictors. In Random Forest, each random tree only considers a random subset of predictors out of which the one that gives the best split is chosen. Random Forest is easy to understand as it has just 2 main parameters.

- Number of trees to be generated
- Number of random features (predictors) for each random tree. The random trees produced are merged into one result using voting for classification and average using regression.

We deemed that this is suitable for CTR problem due to following characteristics.

- Multi-dimensionality: There are 40 features in our problem. Hence running bagging with decision trees is computationally very expensive. As we can limit the number of features to be used for each random tree, Random forest does seem like a natural choice.
- Dataset size: Technical literature mentions that Random Forests are 'embarrassingly parallel'. The training dataset that we have has 45 million rows. And hence it is not very convenient to load this data in any data learning tool on a single machine due to obvious memory constraints. Hence, the level of parallelism that can be achieved with Random Forest comes as a blessing and a Map Reduce solution may be implemented in future for more effective analysis.
- Probabilistic Output: Additional to predicting a Click or No-Click, we also need to compute the probability of these events to evaluate the model using Log Loss.

Implementation of random forest using sklearn package in python enabled us to calculate each class label probability.

B. *Logistic Regression*

Logistic Regression is a statistical classification model which uses probabilistic approach to predict the required values. It can be used to predict the result of a categorical dependent variable also termed as class label using one or more independent variables. A logistic or logit function is used to determine the probability of a single trial.

More often, Logistic Regression is specifically used where the dependent variable has binary outcomes. Sometimes, the dependent variables may have multiple possible outcomes. In those cases, we use multinomial logistic regression. It is a special case of generalized linear model and we can thus say that it is analogous to linear regression. However, we prefer logistic regression over linear regression to confine the output sigmoid function between 0 and 1 whereas in linear regression the output might go over or below this range.

The Methodology:

The logistic model computes the probability of the dependent variable as a function of the values of the predictor variables. The predictor variables are always continuous in the logistic formula. If the initial categorical variables are not continuous, then some preprocessing is required to make them as continuous variable. The logistic function or sigmoid function is as follows:

$$F(x) = \frac{1}{1 + e^{-(\beta 0 + \beta 1 * x1 + \cdots + \beta k * xk)}}$$

Where β0 is a constant and βi are coefficients of the predictor. The computed value, P, is a probability in the range 0 to 1. The logistic function is useful because it can take an input with any value from negative infinity to positive infinity and the output is given in terms of probability between 0 and 1.

The regression coefficients are calculated using the log likelihood estimation. The log likelihood of the model is the value that is maximized by the process that computes the maximum likelihood value for the βi parameters. It is notable that the gradient is quite similar to the linear regression gradient

C. *Model Evaluation Method*

We are using Log Loss in this case as we have a-priori information that our class is either 0 or 1. So, the conditional probability used in log Loss give the distribution on {0, 1} unlike say in Quadratic Loss where the Gaussian assumption gives distribution on all the real numbers. Moreover, we didn’t choose 0-1 Loss due to over fitting of data that happens in it and which results in not so accurate prediction of new data.

Suppose y is the response for reference x, then the log loss is the negative expected log probability of the response given the reference, namely

$$LogLoss = -\frac{1}{n}\sum_{i=1}^{n}(x_i\log(y_i) + (1-x_i)\log(1-y_i))$$

Since our reference is 0 or 1 in this problem, this will reduce the log loss definition into probability classifier

IV. EXPERIMENTS

A. *Exploratory Data Analysis*

The following table describes the data type of all the columns in dataset.

TABLE 1 COLUMN IN DATASET

Attribute	Data Type	Comment
ID	Nominal	Inserted to get row IDs
Label	Binary Class	Describes click or no-click event
I1 through I13	Integer	Numerical features
C1 through C26	Categorical	Hashed categorical features

The semantic names of columns are not provided and categorical variables are hashed for the purpose of anonymizing user data. But we can say that all of the features fall into one of the following categories.

- Context features: These are the features of the web page or of publisher on which and the ad is shown.
- Content features: These are the characteristic details of the advertisement and its Meta data.
- User features : As the name suggests, these are the properties of the user
- Feedback features: This can contain the aggregated history data of this transaction.

Graphs for Class Label and Attributes C1 and C9

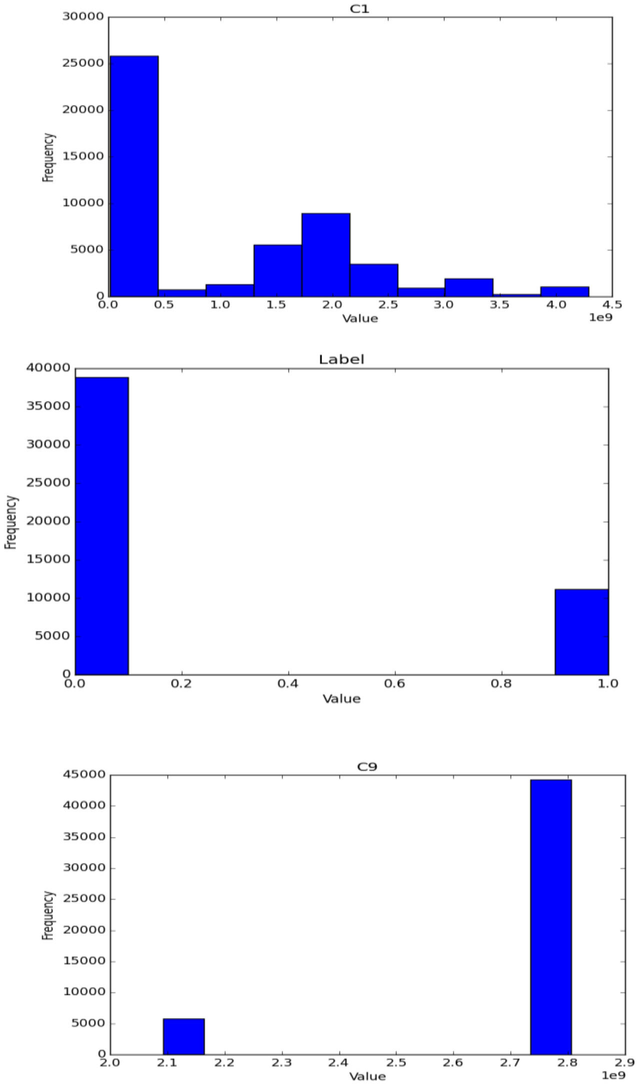

TABLE 2 INTEGER ATTRIBUTE DETAILS

Attribute	Max Value	Min Value	Mean	Std. deviation	Sparsity (%)	Distinct Values
I1	556	0	3.611	9.99	45%	123
I2	18522	-2	111.8	409.32	0	2033
I3	65535	0	38.93	552.18	19%	676
I4	417	0	8.33	11.07	19%	107
I5	1618112	0	17399.5	64725.78	5%	15181
I6	13560	0	141.76	363.36	25%	1621
I7	8807	0	14.75	65.75	5%	479
I8	4352	0	13.32	37.14	0	107
I9	12661	0	123.99	274.72	5%	1530
I10	6	0	0.6	0.67	45%	7
I11	101	0	2.35	4.58	5%	75
I12	493	0	0.95	6.2	77%	57
I13	4016	0	11.31	39.37	19%	213

B. *Preprocessing*

To deal with the high dimensionality, sparsity and large quantity of data we have taken the following pre-processing steps.

We have converted all the categorical columns having 8 character hashed values to integers by using base 16 for efficient memory usage.

Many datasets of real world have missing values. These missing values have to be handled as a part of preprocessing using a suitable method. The easiest way is to eliminate the entire column or row containing missing data which can result in loss of important data. And so in our problem we have substituted the missing value of integer columns by the median of that attribute and the missing values of categorical attribute by the mode value.

TABLE 3 CONFUSION MATRIX: RANDOM FOREST

0	**1**	**< = Classified as**
37090	1709	0
9220	1980	1

TABLE 4 CONFUSION MATRIX: LOGISTIC REGRESSION

0	**1**	**< = Classified as**
38258	541	0
10551	649	1

TABLE 5 DIFFERENT MEASURES

Measures	**Random Forest**	**Logistic Regression**
Precision	0.742	0.73
Recall	0.781	0.778
F-Measure	0.736	0.701
ROC Area	0.687	0.699
TP Rate	0.781	0.778
FP Rate	0.649	0.734

TABLE 6 COMPARISION ON EVALUATION PARAMETERS

Evaluation Parameter	**Random Forest**	**Logistic Regression**
Total number of instances	49999	49999
Correctly Classified Instances	39070	38907
Incorrectly Classified Instances	10929	11092
Accuracy	78.14%	77.81%
Mean Absolute Error	0.3037	0.3196
Root mean squared error	0.4041	0.3992
Kappa statistic	0.1743	0.0645
Relative absolute error	87.34%	91.91%
Root relative squared error	96.93%	95.75%

C. *Results*

The results for Random Forest and Logistic Regression are tabulated in the above tables.

V. CONCLUSION AND FUTURE SCOPE

In this problem we have used Logistic Regression and Random forest. From the results we have got that Random Forest achieved more accuracy than Logistic Regression.

For solving this Big Data problem, the most challenging aspect is to manage scale of the experimented approach of Random Forests and Logistic Regression on the whole dataset.
Another challenge is to represent and modeling inter-feature relationship of categorical variables. This gives us following alternatives.

- Parallelization of Random Forests

We can implement random forest in parallel like using random jungles which will work well with high dimensional data too. Random Jungles can be very useful when we have multiple CPUs, it can implement random forest using multi-threading and Message Parsing Interface (MPI) parallelization on each CPU.

- Using mini-batch logical regression

We can implement logical regression framework that uses a feature hashing to split each categorical variable having k values to k binary variables and then mapping each of the categorical variable in same space to perform logical regression. Other advantage of the framework is its impressive memory footprint. We only need to keep the batch of training data in memory on which the model is trained on. This comes as a boon in comparison to memory inefficient implementations of the logistic regression algorithm in Weka. This may make solving this problem realistic on a humble hardware configuration.

REFERENCES

[1] Thore Graepel ,Joaquin Quiñonero Candela ,Thomas Borchert , Ralf Herbrich, "Web-Scale Bayesian Click-Through Rate Prediction for Sponsored Search Advertising in Microsoft's Bing Search Engine," Mathematical Problems In Engineering.

[2] Sculley, D,"Web-Scale K-Means Clustering," Proceedings Of The 19th International Conference On World Wide Web, pp. 1177 - 1178.

[3] Gupta, Chetan; ”GenIc: A Single Pass Generalized Incremental Algorithm for Clustering,” Grossman, Robert Society For Industrial And Applied Mathematics. Proceedings Of The Siam International Conference On Data Mining, p. 147.Journal Article

[4] He, Dengcheng; Zhou, Yongluan; Chen, Gang; Shou, Lidan,” Cluster Based Rank Query over Multidimensional Data Streams,” Proceeding Of The 18th Acm Conference On Information And Knowledge Management, pp. 1493 - 1496

[5] Ilya Trofimov , Anna Kornetova , Valery Topinskiy ,“Using boosted trees for click-through rate prediction for sponsored search” Proceedings Of The Sixth International Workshop On Data Mining For Online Advertising And Internet Economy, pp. 1 - 6.

[6] Chen, Ye; Yan, Tak,” Position-normalized click prediction in search advertising,” Proceedings Of The 18th Acm Sigkdd International Conference On Knowledge Discovery And Data Mining

[7] Carciolo, M, "Machine Learning with Python - Logistic Regression". Retrieved January , 2014 Available: Machine Learning with Python - Logistic Regression.

[8] Koenig, Inke R; Ziegler, Andreas; Schwarz, Daniel F, “On safari to Random Jungle: a fast implementation of Random Forests for high-dimensional data”,Bioinformatics 26.14