**MALAVIYA NATIONAL INSTITUTE OF TECHNOLOGY,**

**JAIPUR**

****

A

B.Tech DISSERTATION Report

on

**CLICK THROUGH RATE PREDICTOR MODEL**

Bachelor of Technology

(Computer Science and Engineering)

in

Department of Computer Science and Engineering

(2012-2016)

**Submitted By** :

Aditya Choudhary (2012UCP)

**Supervisor** : Bhanu Pratap (2012UCP)

Dr. Pilli Emmanuel Shubhakar Dipankar Bhardwaj(2012UCP1607)

****

Department of Computer Science and Engineering

**Malaviya National Institute of Technology , Jaipur**

**CERTIFICATE**

This is to certify that the Dissertation Report on "**Click Through Rate Predictor Model**" by **Aditya Choudhary, Bhanu Pratap and Dipankar Bhardwaj** is the work completed under my supervision, hence approved for submission in fulfillment of Research Project.

(**Dr.Pilli Emmanuel Shubhakar**)

Assistant Professor

Place: Department of Computer Science and Engineering

Date : May 2016 MNIT,Jaipur

DECLARATION

We, Aditya Choudhary, Bhanu Pratap and Dipankar Bhardwaj, declare that this report titled, "Click through Rate Predictor Model" and the work presented in it written by ourselves. We confirm that:

* This work is done towards the fulfilment of the dissertation report at MNIT, Jaipur.
* Where we have consulted the published work of others, this is always clearly

attributed.

* Where we have quoted from the work of others, the source is always given.
* We have acknowledged all main sources of help.

Signed:

Date:

ACKNOWLEDGEMENTS

We would like to give sincere thanks and gratitude to our esteemed supervisor

Dr. Pilli Emanuel Shubhaker (Department of Computer Engineering, Malaviya

National Institute of Technology, Jaipur) for providing his valuable guidance and

encouragement to learn new things and enhancing our knowledge in the field of

Click Through Rate. His kind cooperation and suggestions throughout

the course of this research guided us with an impetus to work, and successfully

complete the project.

**CONTENTS**

**Abstract i**

**List of Figures ii**

**1 Introduction 1**

* 1. Objective and Motivation…………………………………………………
  2. Related Work………………………………………………………………

1. **Dataset**

2.1 Introduction to dataset………………………………………………………

2.2 New features………………………………………………………………..

2.3 Categorical Variables……………………………………………………….

2.4 Hashing Trick………………………………………………………………..

1. **Classification Models**
   1. Naïve Bayes ………………………………………………………………
   2. Logistic Regression……………………………………………………….
   3. Logistic Regression with Regularization……………………………….
   4. FTRL ………………………………………………………………………..
2. **Experimental Results and Discussions**
   1. Comparison of Naïve Bayes on datasets of different size……………
   2. Comparison of Logistic Regression on datasets of different size……
   3. Comparison of Logistic Regression and FTRL ………………………..
   4. Comparison of Naïve Bayes and Logistic Regression on Spark….
3. **Conclusion and Future Work**

**6 Bibliography**

**ABSTRACT**

Ad click-through rate (CTR) prediction is to estimate CTR with click log, which is influenced by the page information, the position, the user properties, the nature features of ad and some other factors. The right ads for the query and the order they are displayed greatly affects the revenue the company receives from these ads. Therefore, it is important to be able to estimating CTR precisely with click log in sponsored search advertising system. We implemented useful CTR prediction model namely Naïve Bayes, Logistic Regression and FTRL for ads of abundant history data. We also show different results obtained on these models on different sizes of datasets and on different machines.

**LIST OF FIGURES**

**Chapter 1**

**Introduction**

* 1. **Objective and Motivation**

Most major search engines today are funded through textual advertising placed next to their search results. The market for these search advertisements (sometimes referred to as “paid search”) has exploded in the last decade to $5.75 billion, and is expected to double again by 2010 [17]. The most notable example is Google, which earned $1.63 billion in revenue for the third quarter of 2006 from search advertising alone [2] (a brief summary of the history of sponsored search can be found in [7]). Though there are many forms of online advertising, in this paper we will restrict ourselves to the most common mode: pay per performance with a cost-per-click (CPC) billing, which means the search engine is paid every time the ad is clicked by a user (other models include cost-per-impression, where advertisers are charged according to the number of times their ad was shown, and cost per-action, where advertisers are charged only when the ad display leads to some desired action by the user, such as purchasing a product or signing up for a newsletter). Google, Yahoo, and Microsoft all primarily use this model.

To maximize revenue and user satisfaction, pay-per-performance systems must predict the expected user behavior for each displayed advertisement and must maximize the expectation that a user will act (click) on it. The search system can make expected user behavior predictions based on historical click-through performance of the ad. For example, if an ad has been displayed 100 times in the past, and has received 5 clicks, then the system could estimate its click-through rate (CTR) to be 0.05. This estimate, however, has very high variance, and may only reasonably be applied to ads that have been shown many times. This poses a particular problem when a new ad enters the system. A new ad has no historical information, so its expected click-through rate is completely unknown. We address the problem of estimating the probability that an ad will be clicked on, for newly created ads and advertising accounts. We show that we can use information about the ad itself (such as the length of the ad and the words it uses), the page the ad points to, and statistics of related ads, to build a model that reasonably predicts the future CTR of that ad.

The key task for a search engine advertising system is to determine what advertisements should be displayed, and in what order, for each query that the search engine receives. Typically, advertisers have already specified the circumstances under which their ads may be shown (e.g., only for certain queries, or when certain words appear in a query), so the search engine only needs to rank the reduced set of ads that are matches.

As with search results, the probability that a user clicks on an advertisement declines rapidly, as much as 90% [5], with display position (see Figure 1). Thus, it is most beneficial for the search engine to place best performing ads first. Note that, because the probability of clicking on an ad drops so significantly with ad position, the accuracy with which we estimate its CTR can have a significant effect on revenues.

The number of eligible advertisements matching a given query usually far exceeds the number of valuable slots. For example most users never go beyond the first page of search results, in which case the number of ads displayed is limited to the set shown on the first page (this number tends to range between 5 and 8 for the most common search engines). Even within the first page, the significant decrease in CTR by ad position means that ads in very low positions have less impact.

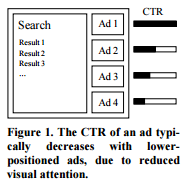
In order to maximize ad quality (as measured by user clicks) and total revenue, most search engines today order their ads primarily based on expected revenue:

(The most notable exception to this is Yahoo, which orders ads based on advertiser bid alone, but plans to switch to using expected revenue soon). The CPC for an ad is its bid (in a first price auction) or the bid of the next-highest bidder (in a second-price auction), optionally normalized by ad performance. The details of the relation between CPC and bid are not important to this paper, but are the study of many works on search engine auction models [8][12].

Thus, to ideally order a set of ads, it is important to be able to accurately estimate the p(click) (CTR) for a given ad. For ads that have been shown to users many times (ads that have many impressions), this estimate is simply the binomial MLE (maximum likelihood estimation), #clicks / #impressions. (In this paper, we assume that over time each ad converges to an underlying true click-through rate. We ignore ads that exhibit periodic or inconsistent behavior for the purposes of this paper, although the work could be extended to such cases.) However, because the CTR for advertisements is relatively low, the variance in this estimate is quite high, even for a moderate number of impressions. For example, an ad with a true CTR of 5% must be shown 1000 times before we are even 85% confident that our estimate is within 1% of the true CTR. In general search advertising, the average clickthrough rate for an ad is estimated to be as low as 2.6% [4].

The time over which the system converges reflects a large amount of search monetization. For example, an ad with a cost per click of $1.60 (an average rate on Google [4]) would require $80 of click through behaviour to experience 50 clicks. Any error in the click through rate estimation during that time will result in suboptimal ranking and thus lost revenue for the search engine and lower traffic for the higher performing ads.

The search advertising market has grown significantly in recent years; there are many new advertisers that enter the market each day. Simultaneously, existing advertisers frequently launch new advertising campaigns. Many advertisers create new campaigns each month, some even every day; others create side-by-side orders for testing purposes in order to optimize their ad performance. All of these practices result in an increasing number of ads to be ranked for each query.



Additionally, existing ads are sometimes targeted to new queries. Some advertisers attempt to increase their return on investment by targeting thousands of infrequently searched terms. There has been a significant increase in keyword volume for PPC campaigns: In one study, the number of keywords per campaign per month increased from 9,100 in September 2004 to 14,700 by March of 2005, and was expected to grow to as many as 17,300 by September 2005 [4].

As a result, there is a large inventory of ads for which the search engine has no prior information. These ads need to be ranked with other, already established ads. An incorrect ranking has strong effects on user and advertiser satisfaction as well as on the revenue for the search engine. Thus, for ads that are new, or have not been shown enough times, we must find a way to estimate the CTR through means other than historical observations.

* 1. **Related Work**

**Chapter 2**

**Feature Engineering**

**Chapter 3**

**Classification Algorithms**

**3.1 Naïve Bayes Classifier**

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of [feature](https://en.wikipedia.org/wiki/Feature_vector) values, where the class labels are drawn from some finite set. It is not a single [algorithm](https://en.wikipedia.org/wiki/Algorithm) for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is [independent](https://en.wikipedia.org/wiki/Independence_(probability_theory)) of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible [correlations](https://en.wikipedia.org/wiki/Correlation_and_dependence) between the color, roundness and diameter features.

The Naive Bayesian classifier is based on Bayes’ theorem with independence assumptions between predictors. A Naive Bayesian model is easy to build, with no complicated iterative parameter estimation which makes it particularly useful for very large datasets. Despite its simplicity, the Naive Bayesian classifier often does surprisingly well and is widely used because it often outperforms more sophisticated classification methods.

We implemented the naïve bayes model considering all the 21 features,however it might not work well for clicks data because certain feature like location where ad is displayed,device on which user sees the ad are clearly related to each other.But we wanted to implement the Naïve Bayes to see if our understanding about the data given to us is right.

For this approach we assumed the entire data to be categorical and calculated the counts for all the unique values in each feature with respect to whether the ad is clicked or not. Once we had the counts, the probabilities for the test data were calculated as follows:

|  |
| --- |
|  |
| http://www.saedsayad.com/images/Bayes_rule.png |
|  |
| * *P*(*c|x*) is the posterior probability of *class* (*target*) given *predictor* (*attribute*). * *P*(*c*) is the prior probability of *class*. * *P*(*x|c*) is the likelihood which is the probability of *predictor* given *class*. * *P*(*x*) is the prior probability of *predictor*. |
|  |
| *Example*: |
| The posterior probability can be calculated by first, constructing a frequency table for each attribute against the target. Then, transforming the frequency tables to likelihood tables and finally use the Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction. |
| http://www.saedsayad.com/images/Bayes_3.png |

Advantages of Naïve Bayes Classifier

* Easy to implement.
* Requires a small amount of training data to estimate the parameters.
* Good results obtained in most cases.

Disadvantages of Naïve Bayes Classifier

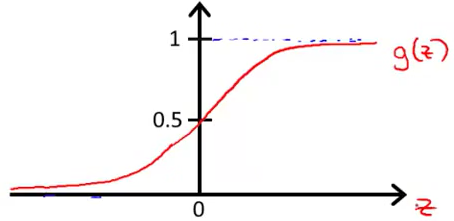
* Assumption of class conditional independence leads to loss of accuracy.
* Practically, dependencies exist among variables. These dependencies cannot be modelled by naïve bayes classifier.

**3.2 Logistic Regression**

Logistic regression is widely used to model the outcomes of a categorical dependent variable. For categorical variables it is inappropriate to use linear regression because the response values are not measured on a ratio scale and the error terms are not normally distributed. In addition, the linear regression model can generate as predicted values any real number ranging from negative to positive infinity, whereas a categorical variable can only take on a limited number of discrete values within a specified range. Logistic regression has proven to be one of the most versatile techniques in the class of generalized linear models. Whereas linear regression models equate the expected value of the dependent variable to a linear combination of independent variables and their corresponding parameters, generalized linear models equate the linear component to some function of the probability of a given outcome on the dependent variable. In logistic regression, that function is the logit transform: the natural logarithm of the odds that some event will occur. In linear regression, parameters are estimated using the method of least squares by minimizing the sum of squared deviations of predicted values from observed values. This involves solving a system of N linear equations each having N unknown variables, which is usually an algebraically straightforward task. For logistic regression, least squares estimation is not capable of producing minimum variance unbiased estimators for the actual parameters. In its place, maximum likelihood estimation is used to solve for the parameters that best fit the data.

Probability of positive outcome

Sigmoid Function

****

Cost Function

**Gradient Descent Algorithm**

Gradient descent is one of the most popular algorithms to perform optimization and by far the most common way to optimize neural networks. At the same time, every state-of-the-art Deep Learning library contains implementations of various algorithms to optimize gradient descent (e.g. [lasagne's](http://lasagne.readthedocs.org/en/latest/modules/updates.html), [caffe's](http://caffe.berkeleyvision.org/tutorial/solver.html), and [keras'](http://keras.io/optimizers/) documentation). These algorithms, however, are often used as black-box optimizers, as practical explanations of their strengths and weaknesses are hard to come by.

Gradient descent is a way to minimize an objective function J(θ)J(θ) parameterized by a model's parameters θ∈Rdθ∈Rd by updating the parameters in the opposite direction of the gradient of the objective function ∇θJ(θ)∇θJ(θ) w.r.t. to the parameters. The learning rate ηη determines the size of the steps we take to reach a (local) minimum. In other words, we follow the direction of the slope of the surface created by the objective function downhill until we reach a valley.

**=**

Repeat {

(Simultaneously update all )

}