A KAGGLE COMPETITION

Predict click through rate

Team members: Arafat Gabareen Mokhtar, Ankesh Tyagi, Dimple Sharma, Rodrigo Zuniga, Vaibhav Chaudry

11/20/2014

Contents

1. The Problem Definition

This proect is a kaggle competition project which has a goal to develop a model to predict click through rate of an advertisement displayed on a webpage. For this research competition, Criteo Labs company has shared a week's worth of click data.

What is your research question?

The goal of this research is to identify a model to predict ad click through rate. Given a user and the page he is visiting, what is the probability that the user will click on a given advisement. This problem is identified as a classification problem which has two classes 0 and 1. If the user did not click on the link the actual value will be 0, whereas if the user clicked on the link then the actual value will be 1. The CriteoLabs dataset contains 40 variables which contain 13 integer variables, 26 categorical variables, and 1 label.

This paper aims to use various machine learning methods learnt in class to develop a model to predict click through rate for advertisements displayed on a website. This paper will attempt to identify a good machine learning algorithm that has a low error rate and best accuracy.

Why this is an interesting question to ask?

Display advertisement is a one of the key uses of machine learning on internet. Online advertisement is a billion dollar industry so the method of predicting a click through rate for advertisements is unknown because they kept under a lock and key. Another reason also is that there is a lack of availability of a good dataset on the internet.

Why we would care about the answer to this question?

Click through rate (CTR) is a way to measure the success of an online advertisement campaign for a website. Online advertisement is a billion dollar industry which is expected to grow more. Large companies like google and Microsoft have used CTR for sponsored search advertisement. CTR can be used in many ways like contextual advertising, display advertising, and real time auction.

A supervised learning model can help to predict CTR based on historic data hence predicting the future success of an advertisement campaigns. CTR predictions can be used to increase convergence rate for an advertisement by displaying targeted advertisements to users based on user preferences. A higher convergence rate of click through rate will result in higher return on advertisement campaign.

Is there any existing research work has tried to answer the same or a similar question?

Some research papers have performed analysis and developed models to predict click through rate for an advertisement. Some large companies like Microsoft and Google have published research paper on click through rate. Following are a few links:

Research Title	Company	Link
Web-Scale Bayesian Click-Through Rate Prediction for Sponsored Search	Microsoft	http://research.microsoft.com/pubs/122779/AdPredictor%20ICML%202010

Advertising in Microsoft's Bing Search Engine		%20-%20final.pdf
Ad Click Prediction: a View from the Trenches	Google	http://static.googleusercontent.com/media/research.google.com/en/us/pubs/archive/41159.pdf
Predicting Clicks: Estimating the Click-Through Rate for New Ads	Microsoft	http://www2007.org/papers/paper784.pdf

2. Approach

Motivation

We were intrigued by the ways companies predict and develop successful web advertisement campaigns. Click Through Rate (CTR) is an interesting problem to look at in the current times when online advertisements are taking over from other conventional mediums.

We were motivated to gain insights from user visit data about their likelihood of clicking on the advertisement.

Explain your methods with sufficient details

We approached this problem from context of the learnings we have had in the class. As a team we decided to apply 5 different models and compare the results to determine which model can predict the Click Through Rate best.

Given the limitations of our computing systems and enormous amount of data; we sampled 20 thousand records (complete records without any missing values).

Data Exploration

With data exploration as our first step after sampling we did several steps in order to better understand the data. Plots and data summaries were used to understand the spread and characteristics of data.

With data exploration we could find out about the factor attributes that had too many levels and were a hindrance to fitting models like Random Forest. We decided to rule out any attribute which had more than 10 factor levels. After removing the attributes with too many levels we used step function and logistic regression technique to find out which attributes were significant in making accurate prediction.

Step function in R helped us in finalizing the attributes that we used in our prediction models, all but one numerical attributes were found to be significant along with 5 factor attributes. Hence, effectively we used 18 predictor variables and one response variable.

After finalizing the attributes to be used we set out to fit different models. The models that we applied are: -

1. Logistic Regression
2. Support Vector Machines

3. Random Forest, fitted the model with 500 trees and also checked whether accuracy increases with increase in number of trees
4. Decision Trees
5. Ensemble Methods (Using decision trees and ridge regression)

3. Data Pre-Processing

Data Source

The dataset consists of one week of click through data provided by Criteo Labs for the Kaggle competition. The original dataset has 40 million rows and 40 variables. There are one dependent/response variable, 13 independent numerical variables and 26 independent categorical variables in the dataset. All levels in categorical variables are encrypted.

Sample

We created several different random samples using the first two million records from the original dataset. We used SAS to extract the first two millions from the original 40 million records dataset. We are aware the sample might not be fully representative of the overall dataset especially if the data is sorted by date but we were not able to load the complete original dataset. We latter also used the readLines() procedure to extract one out of each 10 records and create a different dataset with the extracted records (see appendix for code).

Missing values and number of levels:

The table below shows the percentage of missing values for each variable and the number of levels for the categorical variables. 23 of the independent variables contain missing values with some variables with high prevalence of missing values (V13, V37, V34, V35 and V40).

Several of the categorical variables have thousands of different levels (V18, V19, V22, V25, V26, V27, V28, V30, V31, V33, V39) which is a problem with most of the statistical procedures we want to apply.

15K Sample before imputation and levels reduction:

Variable	Num Nas	PCTN_NAs
V1	-	0
V2	6,453	43.02
V3	-	0
V4	3,256	21.71
V5	3,259	21.73
V6	443	2.95
V7	3,403	22.69
V8	642	4.28
V9	10	0.07
V10	642	4.28
V11	6,453	43.02
V12	642	4.28
V13	11,660	77.73
V14	3,259	21.73

Variable	Levels	PCTN_Blank
V16	214	0.0
V17	434	0.0
V18	8,168	3.2
V19	5,617	3.2
V20	68	0.0
V21	9	11.5
V22	4,051	0.0
V23	115	0.0
V24	3	0.0
V25	4,095	0.0
V26	2,505	0.0
V27	7,607	3.2
V28	2,027	0.0
V29	24	0.0
V30	2,598	0.0
V31	6,747	3.2
V32	9	0.0
V33	1,408	0.0
V34	646	46.5
V35	4	46.5
V36	7,241	3.2
V37	8	75.8
V38	13	0.0
V39	3,364	3.2
V40	39	46.5

Level Reduction and Data Imputation

In order to reduce the number of levels in the categorical variables, the 9 levels with the highest frequencies were identified and remained unchanged. The remaining levels were grouped in a new level with value “ELSE”. If for a given variable fewer than 9 levels accounted for 90% or more of the observations those levels remained unchanged and the remaining of the observations were grouped allowing for further level reduction with variables with less than 9 unique levels. Different number of levels was tried.

To impute missing values for categorical variables we treated those missing values just as another level. For numerical variables we imputed the missing values using the median of the non-missing values which, unlike the mean, has the advantage of being immune to outliers and be fairly easy to compute.

4. Experimental Results

How did you evaluate your solution?

Dataset consists of user visits on a website. It has 40 predictor variables and one response variable. Below are Data acquisition steps:

1. Dataset was first cleansed by removing blank and NAs.

2. Randomly created smaller subset from bigger data set.

3. Analyzed all 40 features and selected 17 most relevant features.

Once clean data was available, we created various models and evaluated them by calculating error rate. Below are the models with their error rate:

Model	**Error Rate**
Logistic regression	0.3
Decision Tree	0.2772
Random Forest	0.282
SVM	0.283
Ridge Regression	0.359

What measures did you use?

The key goal to measure performance of the algorithm is to minimize the prediction error on the test data. Hence to measure the performance of algorithms methods like training data, test data, and cross validation were used for each algorithm with a goal to minimize the prediction error on the test set. For some algorithms the dataset was divided into 70% training data and 30% test data while for many algorithm like Support vector machine, random forest, etc. 10 – fold cross validation techniques were used to evaluate the performance of the algorithm.

The project uses two key methods to measure performance of the classification problem to predict user click – 1 click and 0 no click. Classification accuracy method is used to calculate the predictions error, that is number of record correctly classification divided by total number of responses gives accuracy of records correctly classified. Subsequently, to get the error rate for classification problem the formula deducts the accuracy of correctly classified records from 1. The final result is error rate in decimal which is converted to percentages.

Formula for prediction error = 1 – (no. of records correctly classified/total no. of response)

A second method used to calculate the prediction error is root mean square error.

Formula of root mean square = square root of (sum(y hat - response) / total no. of response)

Methods such as tuning were used by many algorithms to find the best fit model that to solve problems such as over fitting and under fitting the model on the training data. For support vector machine, cost and gamma parameters were used to tune the model to get the best fit model. For decision trees, different tree depth with a binary split or two split was used to tune decision trees to find the best fit model. Finally, random forest algorithm used a range of random forest trees to identify the best fit model.

5. Conclusions and ideas for future work

Conclusion

In summary, a data set on a click through rate was posted on Kaggle.com with the goal to let a free competition to predict the response. The response has two level values of 1 (click) and 0 (not a click). A subsample of 20,000 observations was selected from the entire data set (4 million observations). Several machine learning practices have been studied on the subsample. These methods include Logistic Regression, Support Vector Machine, Ensemble Method, and Decision Trees. All methods were used to predict the outcome (y=0 or y=1). Similar results from these machine learning techniques have been obtained with error rates vary from 0.28 to 0.30, which indicates that no favorite method can be used to classify the subsample data.

Future improvements

The original dataset contains over 4 million observations which is several orders of magnitude larger than what was used in this study. The main reason for analyzing only a subsample is due to the fact that no single machine can analyze the entire dataset. A computer cluster such as Hadoop cluster could be beneficial to serve the purpose of this analysis. In such a case, a more robust classification could be achieved with lower error rate. Moreover, since the outcome is over populated with zero values, the nonzero outcome may treated as anomalies rather than classifications.

6. References

1. https://www.kaggle.com
2. The dataset for click through rate was provided by CriteoLabs: http://labs.criteo.com/downloads/2014-kaggle-display-advertising-challenge-dataset/
3. Lecture Notes and resources from Patricia J Hoffman
4. R packages from **cran**.**r**-project.org/
5. For problem solving errors **stackoverflow**.com/

7. Project proposal and team tasks

Each team member of the team equally contributed towards the project, team discussions, presentation, and the final report. Following are the tasks performed for executing the project.

Task 1. Problem analysis: Analyze the problem statement.

Task 2. Explore data and sampling: In this step, the idea is to explore the dataset and create a sample dataset. The dataset of criteo website is large and cannot be handled with a single machine so the project will use a sample of the data.

Task 3. Feature analysis: Analyze the features of the dataset and perform feature normalization if required.

Task 4. Choose an Algorithm : In this step, the goal is to analyze characteristics of various algorithms and select appropriate model to implement.

Task 5. Implement Algorithm: This step includes implementing the selected algorithms and fit the model to the dataset.

Task 6. Cross Validation: This step includes testing the models with cross validation set.

Task 7. Examine Fit and Update Until Satisfied: Examine whether the model fit the dataset and further refine the model if required.

Task 8. Use Fitted Model for Predictions: Using the model make predictions and test outcome of the prediction.

Task 9. Compare different models and techniques.

The table below describes machine learning algorithms implemented by team members.

Name	**Algorithm implementation**
Arafat Gabareen Mokhtar	Logistic regression
Ankesh Tyagi	Random Forest
Dimple Sharma	Support vector machine
Rodrigo Zuniga	Ensemble method with Decision tree and Linear regression
Vaibhav Chaudry	Decision Trees

8. Appendix

Data pre-processing code

#SAMPLING LARGE FILES:

#Get one out of every 10 lines from a large file and output it to the smpl7 tab delimited text file

#Print row number and number of characters on the selected row

x<-file("~/20140919_ucsc_machine_learning/20141020_HW6/dac.tar/train.txt","r")

```
out2<-file("smpl7.tab","w")
i=0
for (i in 1:50000) {
ln<-readLines(x,n=1)
if (i%%10==0) {print(nchar(ln))
        print(i)
        writeLines(ln,out2)}
}

#REDUCING NUMBER OF LEVELS FOR CATEGORICAL VARIABLES TO A MAXIMUM OF 10 LEVELS:
#IMPUTE MISSING VALUES FOR NUMERICAL VARIABLES USING MEDIAN

s<-read.table("~/20140919_ucsc_machine_learning/20141020_HW6/SAMPLE50K.tab",
header=F,nrows=1000,sep="\t")

#reduce number of levels on categorical variables to a maximum of 10 levels leaving only the 9 most frequent levels and replacing the rest with an "ELSE" level
s2<-matrix(nrow=nrow(s),ncol=26)
lvls<-c()
for (i in 1:26) {
Vx<-as.vector(s[,i+14])
t<-data.frame(prop.table(table(s[,i+14])))
tb<-t[order(t$Freq,decreasing=T),]
tc<-cumsum(tb$Freq)
if (length(tc>0.9)>9) lvl=9 else lvl=length(tc>0.9)
lvls<-c(lvls,tc[lvl])
```

```
td<-as.vector(tb[1:lvl,]$Var1)
print(c("i =",i))
print(length(td))
Vx[!(s[,i+14] %in% td)]<-"ELSE"
Vx[s[,i+14] %in% c("")]<-"BLANK"
table(Vx)
s2[,i]<-Vx
}

s3<-as.data.frame(s2)
colnm<-paste("V",15:40,sep="")
names(s3)<-colnm

s4<-cbind(s[,1:14],s3)
colnm<-paste("V",1:40,sep="")
names(s4)<-colnm

#Replace NAs with median
for (i in 2:14) s4[is.na(s[,i]),i]<-median(s[,i],na.rm=T)

#oversampling: 50% positive and 50% negative:
s4.train<-s4[1:35000,]
s4.test<-s4[35001:nrow(s4),]

s4.train.1<-s4.train[s4.train[,1]==1,]
```

```
s4.train.0<-s4.train[s4.train[,1]==0,]

oversmpld<-rbind(s4.train.1,s4.train.0[1:nrow(s4.train.1),])


t1<-
rpart(factor(V1)~.,data=oversmpld,control=rpart.control(maxdepth=15),method='class',parms=list(prior
=c(.75,.25)))

rpart.plot(t1)
```