**Business Understanding**

It is crucial in advertising industry to make informed decisions on which advertisements to show where and to whom in order to maximize the profit. One way to achieve this is to predict the outcomes (Whether this ad will be clicked or not) using the historical behaviours and patterns as most of the advertisements are pay per click basis.

When it comes to real time bidding, customers in real time would bid for the as slots. From the advertisers’ view the high bid ad with less probability of getting clicked might be better than low bid ad with high probability of getting clicked.

This explains the importance of predicting the click through rate of the advertisements which will enable informed decision making about which ad to display so as to reduce cost and increase profit.

**Data Understanding**

We have 24 columns in the data including the target variable (the variable to predict), click. The data is two files – train.txt and test.txt.

The two files are imported into two tables trainData and testData created in a database named CTR\_Prediction.db in sqlite3 for initial analysis and processing.

**Train.txt data has 24 columns including the target column:** below are the unique values per column

Total number of rows = 2847802

**Distinct values:**

1. Click (Target) : 2
2. Weekday : 7
3. Hour : 24
4. Timestamp : 2434838
5. Log Type : 1
6. User ID : 2641292
7. User-Agent : 36
8. IP : 489690
9. Region : 35
10. City : 370
11. Ad Exchange : 3
12. Domain : 15146
13. URL : 715316
14. Anonymous URL ID : 1
15. Ad slot ID : 51529
16. Ad slot width : 11
17. Ad slot height : 6
18. Ad slot visibility : 4
19. Ad slot format : 3
20. Ad slot floor price (RMB/CPM) : 197
21. Creative ID : 11
22. Key Page URL : 2
23. Advertiser ID : 1
24. User Tags : 720665

**Data Preparation**

The column marked in green, click is the variable to predict which will be the target variable in our analysis.

The columns marked in red, Log type, Anonymous URL ID, Advertiser ID have only one unique value meaning it does not change over samples irrespective of the class in the given data set. So there is no variance and these three variables will not be useful while building a prediction model. Hence the three variables are neglected from the analysis before start.

The columns in yellow, Timestamp, User ID, IP User Tags are almost unique in most of the samples and this will have a tendency to bias the model towards one particular class.

Here in the train.txt we have two classes 0(negative) and 1(positive) which makes our problem a binary classification.

There are only 2076 positive samples and all the remaining samples are negative samples. This makes the dataset unbalanced increasing the risk of overfitting to one majority class (class 0 in this case). The dataset is highly unbalanced approximately in ratio 1:1000 and the yellow marked columns have very high variance almost unique for most of the samples. There are possibilities that the model will learn incorrectly that the class can never be positive if the values of these columns are equal to those values in negative samples, only because no positive samples have those values in training data.

The user tags column has variable number of comma separated tags for each sample. There are many unique tags and many combinations of them in samples. Initially the column user tags is not considered but worth to give a try on transformation and usage of this column.

Although timestamp can be derived into multiple derived columns year, month, date, hour, minute, sec, millisecond which might be really useful variables individually as the behaviour in this industry changes over time, for the initial round of model building all the four very high variance columns are neglected from the analysis. However timestamp makes sense and will be considered in the second round by deriving columns from timestamp.

So far 7 independent variables are eliminated from the analysis with heuristic analysis and data exploration (Timestamp needs to be considered at a later point). The sqlite3 scripts are placed in the repository folder for reference. Now to start analysis, there are 16 variables and 1 target from training data.

1. Weekday : 7
2. Hour : 24
3. User-Agent : 36
4. Region : 35
5. City : 370
6. Ad Exchange : 3
7. Domain : 15146
8. URL : 715316
9. Ad slot ID : 51529
10. Ad slot width : 11
11. Ad slot height : 6
12. Ad slot visibility : 4
13. Ad slot format : 3
14. Ad slot floor price (RMB/CPM) : 197
15. Creative ID : 11
16. Key Page URL : 2

Still the columns domain, url, ad slot id have many unique values but they will be eliminated if required after dimension reduction.

**Modeling and Evaluation (in parallel)**

**Dimension Reduction - Recursive Feature Elimination and Cross Validation**

|  |  |  |  |
| --- | --- | --- | --- |
| CV | No. of folds | Estimator | Scoring |
| StratifiedKFold | 10 | LogisticRegression | Accuracy |



**Variable Importance**

2000 random trees are built to see the variable importance and neglect the least important variables.



With above analysis, the top 12 features from two methods are considered for further analysis.



Three trials each for 5 runs are made using the above table.

1. RFE or Forest = 14 features
2. RFE = 12 features
3. RFE and Forest = 10 features



Since the accuracy increased slightly on every variable reduction cycle, finally the 10 variables present in both RFE and Forest top 12 are considered from now for model building. This number of features can be later modified and tested to increase model accuracy after building the base model with the above selected features with approximately 60 percent accuracy.

***Deep attention given to below points so far***

Since the dataset is highly unbalanced it is must to address this in order to avoid over fitting. Overfitting may lead to high training accuracy because the model will learn to classify any input as negative class so that the accuracy improves.

So maintaining the balance is important while training. This can be achieved by three main ways.

1. Under sampling
2. Over sampling
3. Providing Sample weights

To be simple and easy, under sampling is chosen. The scripts are written in a way that,

1. The count of samples for every class (irrespective of the number of classes)
2. The minimum count of class is chosen
3. Same number of samples are randomly chosen from all other classes to form a final balanced training data.

Under sampling is chosen because in the other two cases the number of training samples in total will be high and an additional problem needs to be solved. (Out of core memory processing).

1. Out of core memory error occurs when all the training samples cannot be kept in memory
2. It can be eliminated though easily by batch processing the samples and storing them in a HDFStorage
3. For now this is not done but it is worth to try all three methods and compare the performance.
4. While batch processing, partial fitting is to be done since the same model needs to be trained incrementally without overwriting. It is the case in most of the online learning problems.
5. CTR prediction is also more kind of online learning problem as the behaviour changes often. Yet the partial training method will be given a try later if time permits.

StratifiedKFold is used to maintain class balance during cross validation.

Encoding of labels in categorical variables

**Deployment**

Logistic Regression model is built with all the samples in the under sampled balanced training data with only the 10 variables selected so far. There are yet model optimizing. This is not the best model yet a model to showcase my skillset.

The trained model had approximately 60 percent accuracy during cross validation consistently.

The built model is used to predict the test.txt which is already dumped in a table called testData in an sqlite3 database names CTR\_Prediction.db

The predicted results are saved as a csv file with name predictions.csv in the submission.

**Further Improvements**

1. In depth look at the left out cases like transformations of raw data timestamp, User Tags, etc.,
2. Further reduction of dimensions along with cross validation of model performance.
3. Trying out different models and different architectures since all the scripts are written in a way that the config file can be modified to try different models and dimensions.
4. Work on speed optimization.
5. Script beautification
6. Increasing autonomy