

X Axis Data Challenge Writeup and Answers

1. Briefly describe your methodology for analyzing the data.

Firstly, I looked at the data statistics, and plotted some variables to see if any of them can distinguish between the clicks and the non-clicks. Most variables had significant overlaps, and the data was very imbalanced, which was one of the challenges. I took a fraction of the background for my analysis (undersampling), performed data transformations (see below for details) and then applied machine learning and optimized my metrics. The following passage below outlines the data description, challenges and analysis steps.

Data Description :

The Data set has the following Columns :

'placement', 'position', 'browser', 'carrier', 'domain', 'supply_type', 'language', 'region', 'os_extended', 'publisher', 'device_model', 'device_type', 'user_day', 'user_hour', 'size', 'response'

Number of events : 159996

Response (0) : **159270 (Non-click)**

Response(1) : **726 (Click)**

Ratio of Response=1(N1)/Response=0(N0) = $0.0046 = 0.45\%$

4. What are some of the challenging things about this kind of dataset? (Other questions addressed later)

Some Challenges:

1. Ratio of clicks/Non-clicks very small : Imbalanced data (See Imbalanced data)
2. Large number of unique domain names – 236
 - a. Device_model = 82
 - b. Publisher = 113
 - c. Placement = 275
3. Categorical variables : size, device_type
4. Correlated variables.
5. No significant difference between variables for clicks vs non-clicks as visualized in plots (Overlapped variables)

Imbalanced data:

The dataset is very imbalanced and would lead to very pure fits. Hence I used a random undersampling of the data to produce a new dataset with a 60/40 ratio of response0/response1. Fraction of (response=0) data sample used = 0.007

New ratio (N1/N0) ~ 65 %

This however reduces the dataset to N1=726, N0=1115

Data Transformation:

To overcome the other challenges, I performed the following data transformations:

1. **Categorical Variables** : Create dummy categories of the categorical variables (drop the original column, and one of the dummies as that would be 100% correlated with the others, so N levels would translate to N-1 columns)
2. **Aggregation and new features** : Aggregate domain names based on number of views per week (See functions, transform, and getdict) and create a new category 'domctr'. I used the variables user_day and user_hour to form this new variable.
3. **Scaling** : I saw a good separation of the publisher variable for the clicks and non-clicks, but the values were very large. Hence I used the 'publisher' variable as a numerical category after scaling, which scaled it to a variable between 1 and -1 with mean 0 (See transform).
4. **Binning** : I transformed the user_day into weekday and weekends (0,1). The user_hour is treated as a continuous variable, though it could also be binned into morning, afternoon, day and night.
5. Apply these transformations to both training and test samples.

Correlations : The most correlated features are as below :

Placement and publisher = 72% ; User_day/supply_type=40% ; Placement/domctr = 16% ; Publisher/domctr = 18% ; Device_model/domctr = 12% ; Device_model/tablet = 42%


Machine Learning:

2. a) Which ML algorithm did you pick for analyzing the data? Why?

Since this is a classification problem, we can use logistic regression or decision trees. I have tried both. A single decision tree suffers from instabilities especially when the data sample is small, hence I have used a randomforests classifier, which splits the data sample and performs N number of decision tree classification, finally aggregating all the trees.

I split the data sample into a training and test sample in a 80/20 ratio.

Please refer to the Jupyter Notebook for the full machine learning results.

DecisionTree Classifier	Predicted 	Negative(0)	Positive(1)
	True	163	67
		40	99

Random Forest	Predicted →	Negative(0)	Positive(1)
	True(0)	179	51
	True(1)	38	101

Logistic Regression	Predicted →	Negative(0)	Positive(1)
	True(0)	193	37
	True(1)	69	70

	Decision Tree	Random Forest	LogisticRegression
False Positives	67	51	37
False Negatives	40	38	69

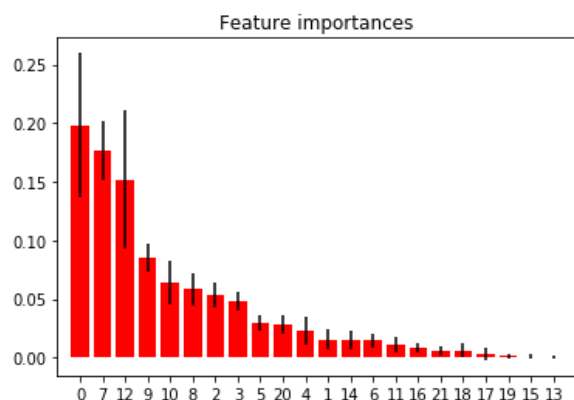
The RF classifier gives the least false Negatives. I picked the RF classifier for the final probabilities. From, the Jupyter notebook, the Precision, Recall and Accuracy values for the RF Classifier are the follows:

Precision: 66.4%

Recall: 72.7%

Accuracy: 75.9%

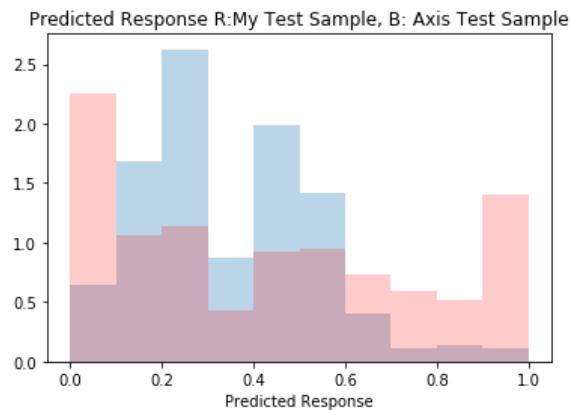
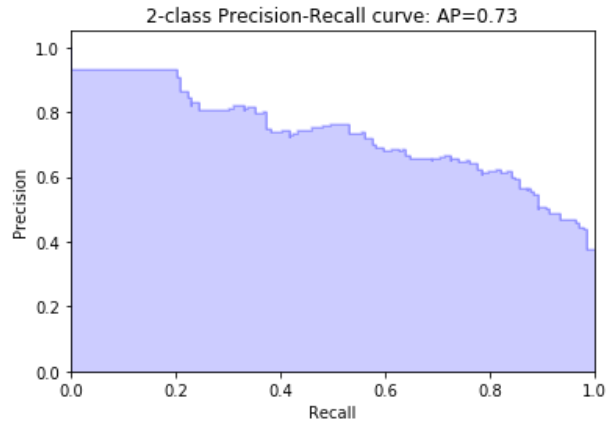
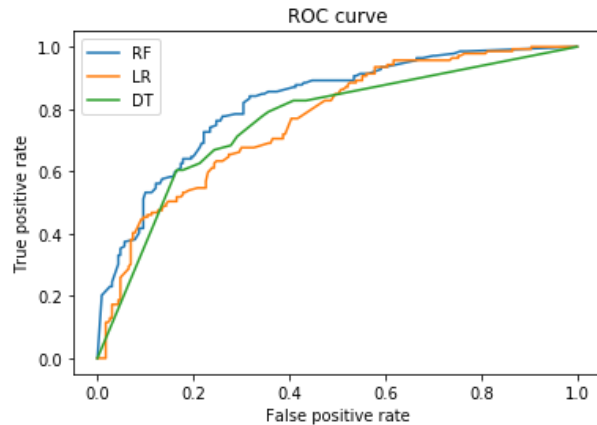
See below for the features Importance from the RF classifier:



Where the features are :

```
'placement', 'position', 'browser',
'carrier', 'domain', 'supply_type',
'language', 'region', 'os_extended',
'publisher', 'device_model',
'device_type', 'user_day',
'user_hour', 'size', 'response'
```

The features in order of importance are: Placement, os_extended, domctr (views/week), device_model, user_day, publisher



b) Which ML algorithm would be a poor choice for this dataset? Why?

A multiple regression algorithm would be a poor choice as the dependant variable has only two choices 0 and 1, while a regression algorithm is used to predict numerical values when the dependant variable is continuous.

3. Which evaluation metric did you pick? Why?

I would try to optimize **recall** (Sensitivity) and **precision** (Specificity) . I would like to reduce false negatives and false positives. Hence, would like to achieve a high precision and a high recall.

Precision (Specificity) : $\text{True positive} / (\text{True positive} + \text{False positive})$

Recall (Sensitivity) : $\text{True Positive} / (\text{Positive} + \text{Negative})$

Accuracy : $(\text{True positive} + \text{True Negative}) / (\text{True positive} + \text{True negative})$

4. This question is answered under methodology (See Challenges)

5. In advertising impression data, which would worry you more: false positives or false negatives? Why?

I would be more worried about a false negative, because I would not want to miss an opportunity to advertise, and classify a potential customer, who would click as a non-click customer. This is under the assumption that the cost of advertising digitally, is potentially much smaller than what the return is. Hence, I would be ok with paying for the ad rather than losing a potential customer. Hence, I would worry more about high false negatives, which would classify a true positive (1) as a 0, than worry about false positives. If money is an issue then I would worry about false positives.

6. Explain regularization to a layperson. What is it and why would you want to use it?

Regularization deals with overfitting the data, and is especially important when we have a small training data set, but we want to predict the probabilities for a much larger test data set. A loss function determines how far our predicted values are from the true values. During overfitting, the model may artificially show a very low loss number. Regularization adds a term to the loss function in order to penalize it in the case of a more complex model as happens during overfitting. We should use regularization In our case, as we have a very small clicks=1 data, and hence we had to undersample the data. But this would lead to overfitting as we reduced the data sample. Overfitting can also be an issue when we have too many unique values in our sample as well, which can't be binned.

7. What's the difference between L1 and L2 regularization methods?

L1 weight regularization penalizes weight values by adding the sum of their absolute values to the error term. Symbolically:

$$\text{Error} = \text{Sum}(\text{Obs.} - \text{True})^2/N + \text{sum}(\text{abs}(w))$$

L2 weight regularization penalizes weight values by adding the sum of their squared values to the error term. Symbolically:

$$\text{Error} = \text{Sum}(\text{Obs.} - \text{True})^2/N + \text{sum}(w^2), \text{ where } w \text{ is the weight and } N \text{ is the number of observations.}$$

8. A big part of a data scientist's job is to explain/summarize the research to other members of the team. Usually that involves using visualizations. Give an example of a good visualization and a bad one. Why are they good/bad?
9. A good visualization would try to visualize all the features and show correlations between them. It would be well labelled. The names of the dependant and independent variables should appear on the axes and the fonts should be visible to the audience. The visualizations should look at the data at various angles, such as through scatter plots, bar plots and histograms.

An example of a bad visualization could be unlabeled plots without titles; scatter plots with too large marker size; Colors not distinguishable in bar plots, or scatter plots. No legends on the plots. A visualization which does not convey the goal is a bad visualization too. Say one is interested in finding the make and model of high priced cars, and one is shown a scatter plot of make vs. model. This would give no information about the value of interest.

10. What is better: 50 small decision trees or one large decision tree? Why?

50 small decision trees is better than one large decision tree because a single decision tree is dependant on the sample, but a sample is never a faithful reproduction of the population. 50 small decision trees however sample the large sample randomly and an aggregated tree would be a more faithful modelling of the population. This is done in adaptive boosting (Adaboost), Xgboost (Extreme Gradient Boosting) or RandomForestClassifier.

Outlook:

- 1) In order to categorize the domains, I also tried to use click through rate, which is the number of clicks to the number of views (can be got from user_hour). It was a powerful variable and gave a big improvement to the machine learning algorithm. However, I found that there were a huge many domain names which appear in the test sample, which would then not benefit from being categorized with the ctr. Hence, instead I decided to use views/week, which is a more weaker variable compared to CTR.
- 2) To improve the specificity and sensitivity one can do more feature engineering and also study different splits in the data sets. Such as 50/50 background/signal; Oversample the signal, and then aggregate the fit parameters over all such samples. That would give a more powerful fit than the present one.