Ad Classification

Deahan Yu

Introduction

These days, ads are everywhere; from TVs and newspapers, and to webpage you want to see. They can be funny at times, and informative in lesser occasions, but most times they hinder you from doing what you want to do. Some ads make themselves even more annoying by making themselves not look like an ad straight up, making you use your brain on unnecessarily, instead of focusing on more useful things. So, it would be nice to have program that tells you whether an image is an advertisement or not by looking at information at hand; what we are trying to build here is exactly that. Given bunch of data that we can gain from meta-data and webpage script, we will try to build a program that predicts whether a picture is an advertisement or not. We will begin by conducting preliminary analysis on the data, using simple statistical techniques. From preliminary analysis for continuous variables, we can see that most pictures in width 400-500 are ads(102 out of 110). Also, after dropping columns that has no 1s, we did some analysis to see if there is noticeable individual relationships between individual columns(apart from col 2 and 3), and found 25 variables that had more than 30 1s and had more than 50% of its 1s were ads. Performing linear regression on those 25 + width gave about 62% R_{sq}; maybe not that bad, but being able to explain just 62% of the variance is probably not enough for any prediction, which is why we should use more developed algorithm than manual checking and multivariate linear regression; we will begin by trying to assess data structure using unsupervised analysis.

Unsupervised Analysis

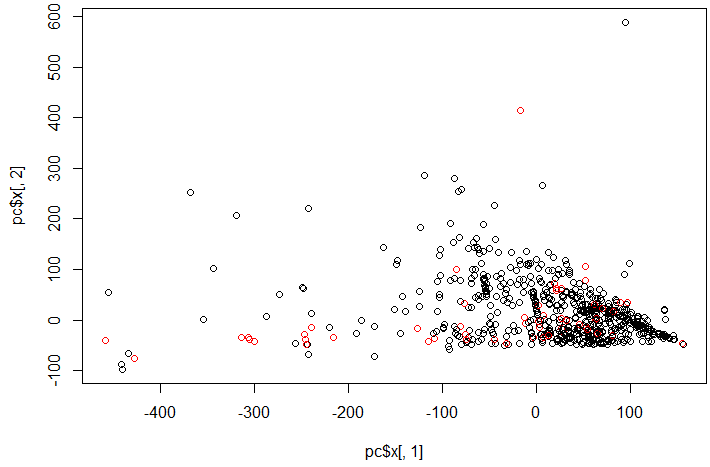

Figure 1Plot of PCA, first 2 scores

Since there were a lot of dimensions and reducing the number would help in understanding the structure, we applied PCA to the dataset and took 2 scores to make it two-dimensional. To check if this is alright, we calculated variance of dimensions from PCA, and it turned out that first 2 scores explain about 77% of the total variance, and amount of variance explained plummeted after that (0.3% on 3rd dimension), so taking 2 scores should reasonably work. To test how good PCA is at representing the

whole data, we ran logistic regression on the data, as well as getting a plot of first 2 scores in PCA. The end result was not satisfactory; 4-fold cross validation (partitioned before PCA) was good at predicting 0s, but not so good at predicting 1s. Error rate for predicting 0 was 0.03, while error rate for predicting 1 was at whooping 0.46. And, as you can see from figure 1, plot did not show noticeable structure.

Next, we tried clustering as a way of finding structure. To find optimal number of clusters that will capture max variance, we used CH index, and got that 3 is the max. Shown in figure 2 is the dendrogram of the complete cluster on training data. From the figure, it is easy to see that the one group has really low number of points. In fact, distribution was (1193,157,9), while the actual number of ads in the training set was 221 out of 1359 observations. Looking into each group, it turns out that group 2 did a good job on capturing ads in that region (118 ads out of 157 total), but it does not capture all the ads in the training set. Points in groups 1 and 3 didn't capture ads or no-ads significantly, showing 102 ads out of 1193 and 1 ad out of 9 respectively.

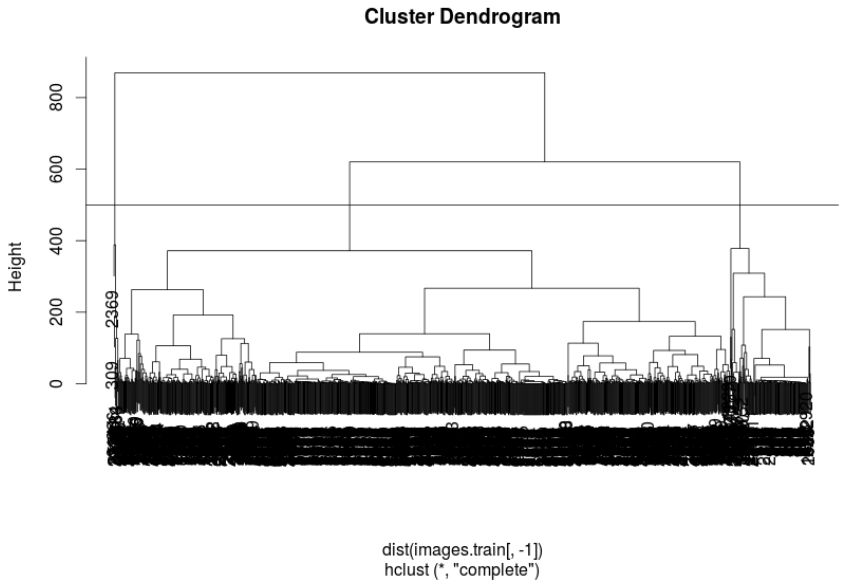

Figure 2 Dendrogram of clustered data using complete linkage

More in-depth analysis of the group yields some other interesting information. For example, take a look at plot of width and height, colored by ad or no ad on left(figure 3), and by groups on the right(figure 4). In figure 3, there are a number of red points between 0 and 200 on both height and width. In figure 4 though, group 2, which refers ad for the most part, is focused on right bottom part only. While the data is from 1416 variables, it looks like clustering took into account only height and width. My attempts at manual shrinking of height and width and PCA didn't improve the prediction.

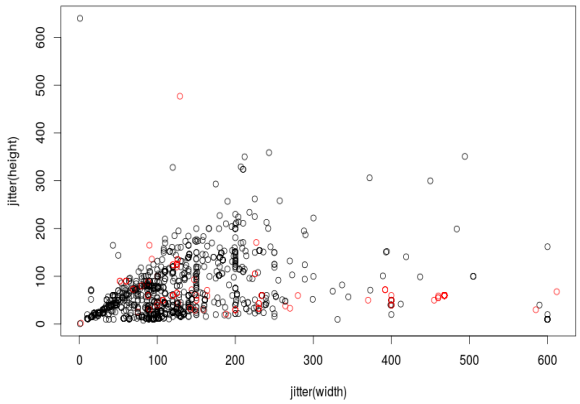

Figure 3Plot of width and height, colored by ad (red) and no ad (black)

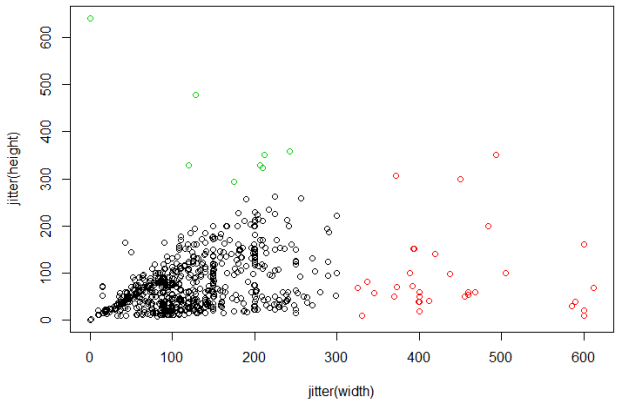

Figure 4Plot of width and height, colored by group3 (green), group2 (red) and group 1(black)

While we discovered some interesting trends in the dataset, such as many ads had width of between 400 and 500, and clustering yielded a group with okay ad prediction that captures about half of total ads in the dataset, we actually need better prediction algorithm to be able to predict whether a picture is an ad or not, which leads to supervised learning.

Supervised analysis

We will discuss how we come up with whether or not each of uncategorized images corresponds to an Ad. While there are many methods for classification, we use random forests classification technique in this report because tree-based techniques run efficiently on large data set. It can also perform well even when there are thousands of variables including binary variables. Random forests gives estimates of what variables are important in a classification as well as unbiased estimate of the test error. In addition, it is model-free. Therefore, we do not have to consider any underlying assumptions for our data set.

Variable Screening on the Training Set

In this section, we will reduce the number of variables as many as possible. As described in the introduction, training data set contains 1558 variables. First, we find out that 142 variables contain only zeros. It is meaningless to keep them in our training set because they are vectors of zeros. Removing those features, our training set has 1416 variables.

Now, we will look at correlation to our response variable and try to eliminate predictor variables, which have low correlation to the response. To determine the best cutoff point, we will compare each out-of-bag MSE estimate for each cutoff that we make. When we choose a correlation cutoff point, we keep variables that are higher than the cutoff point. Then, we run random forests with

those variables and look for an out-of-bag MSE estimate.

Running random forests classification, we do not need to do cross-validation test to get an unbiased estimate of the test error. Instead, we have an Out-Of-Bag estimate of error rate. That is, each tree is constructed using a different bootstrap sample from the training set. For each tree, about one-third of training points are left out. In other words, each sample point has about one-third of the trees that do not include it. We can get an average prediction from the one-third of the trees for each sample point. We combine these predictions over all points to get an out-of-bag estimate of error rate.

For example, if our cutoff point were 0.02, then we would remove any of 1416 variables, which its correlation to the response are lower than 0.02. Our new training set would contain less than 1416 variables. Then, we would run random forests on this training set. This random forests model would give us an out-of-bag MSE estimate of error rate.

Number of variables kept & a cutoff point (Higher than each cutoff point)		Out-Of-Bag estimate of error rate
1416 Variables:	cutoff = 0	2.80%
1284 Variables:	cutoff = 0.02	2.72%
932 Variables:	cutoff = 0.03	2.80%
***595 Variables:**	**cutoff = 0.04**	**2.72%**
484 Variables:	cutoff = 0.05	3.09%
409 Variables:	cutoff = 0.07	3.09%
335 Variables:	cutoff = 0.10	3.02%
82 Variables:	cutoff = 0.20	3.68%

Table 1: *Number of variables at each cutoff for Correlation to the response and out-of-bag MSE estimate calculated by random forests*

Table 1 shows eight different cutoff points, number of variables selected at each cutoff, and its out-of-bag MSE estimate after running random forests. Although the eight cutoff points that we use here are randomly picked, they seem reasonably fair since they cover from a set of 1416 variables to 82 variables. Looking at the table, out-of-bag estimates of error rate start increasing after a cutoff is about 0.04. Until 0.04, out-of-bag MSE estimates are fairly close. If our estimates of error rate were about same, then we would want to pick our pinpoint that has the fewest variables. Therefore, we choose a cutoff point to be 0.04 and keep only 595 variables as a training set. Now, our training set has 595 variables.

Random forests classification on the Training Set

In this section, we will focus on the nature of the relationship between the predictors in our model and the predictions. After variable screening, we run random forests on the training set of 595 variables. We know that random forests gives estimates of what variables are important in the classification.

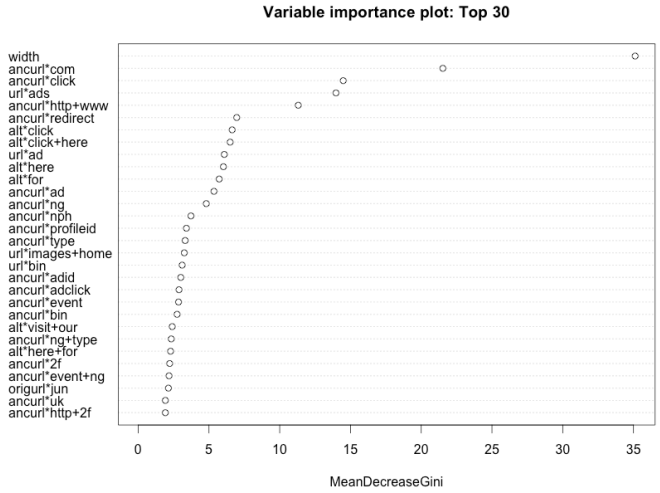

Figure 5: *Plot of top 30 important variables for our random forests model*

Looking at figure 5, we see many variables containing “ad”/“ads” and “click”. These variables are somewhat expected to be important because an advertisement on the Internet needs to be clicked and it should have URL with ad/ads directory.

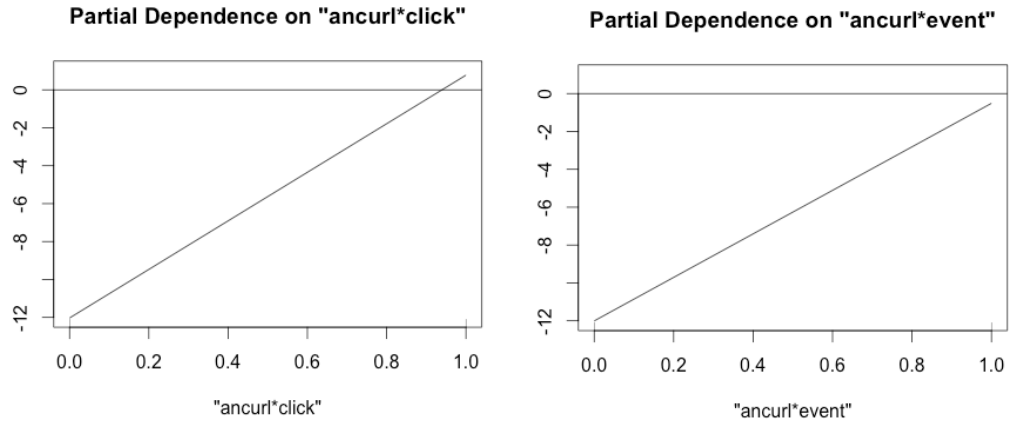

Figure 6: *Partial dependence on “ancurl*click” & “ancurl*event” plots*

Figure 6 shows partial dependence plot on “ancurl*click” variable and “ancurl*event”. “ancurl*click” variable takes third place in the variable importance plot, figure 5 while “ancurl*event” takes 20th place. Knowing that both variables are binary, we clearly see a difference between them. Our accuracy of Ad classification depends on “click” variable. For example, it increases when there is “click”. However, our accuracy of prediction drops when there is “event”.

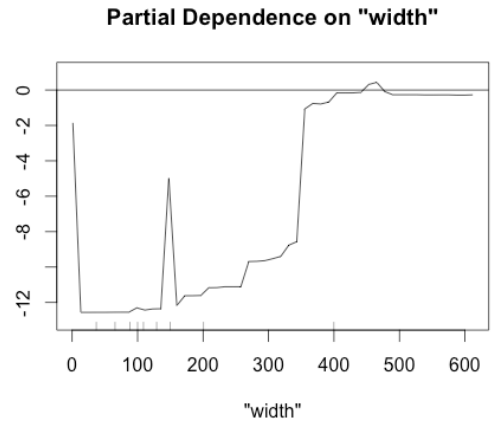

Figure 7: *Partial dependence on "Width" Plot*

Width	Less than 350	More than 350	Total
Ad	103	118	221
Not Ad	1103	35	1138
Total	1206	153	1359

Table 2: *Total actual classes split into two groups*

At the figure 5, we see that the most important variable is "width". Looking at figure 7, our accuracy of Ad prediction increases to positive values when width is between about 450 and 475. On the other hand, we find out an interesting fact. There is a big increase in the accuracy at around 350 despite negative values. When we look at table 2, we see number of Ads is somewhat even between two groups while about 97% of Not Ad is in a group of width less than 350. This shows that most of images that are not ads are less than 350 widths.

Width	0~100	100~200	200~300	300~400	400~500	500~	Total
Number of Ads	27	48	28	4	112	2	221
%	12%	22%	12%	2%	51%	1%	100%

Table 3: *Number of Ads in each range of widths*

In addition, we find out, from table 3, that a width range of 400 and 500 seems standard for advertising on the Internet because 51% (112 out 221) of Ads are in the range. This also tells us why there is a positive effect on the accuracy of the prediction between 450 and 475 at the figure 7.

		Prediction		
		Ad	Not Ad	Class error
True class	Ad	192	29	0.1312
	Not Ad	8	1130	0.0071

Table 4: *Confusion Matrix by random forests on the training set*

Table 4 is a confusion matrix that shows how our random forests predict. Our false negative rate is 13.12%. This tells us that our misclassification rate of predicting Ads would be about 13% if we implemented this model. Our false positive rate is 0.71%, which is really low. This tells us that our model is better at predicting non-Ads than ads.

Predicting the Test Set

We have a random forests model using the training set of 595 variables. We can use predict function in R to make predictions for our test set. Out of 1000 predictions, we have 144 to be ad while 856 to be not ad.