Web Search Classification

Hongyi Xia

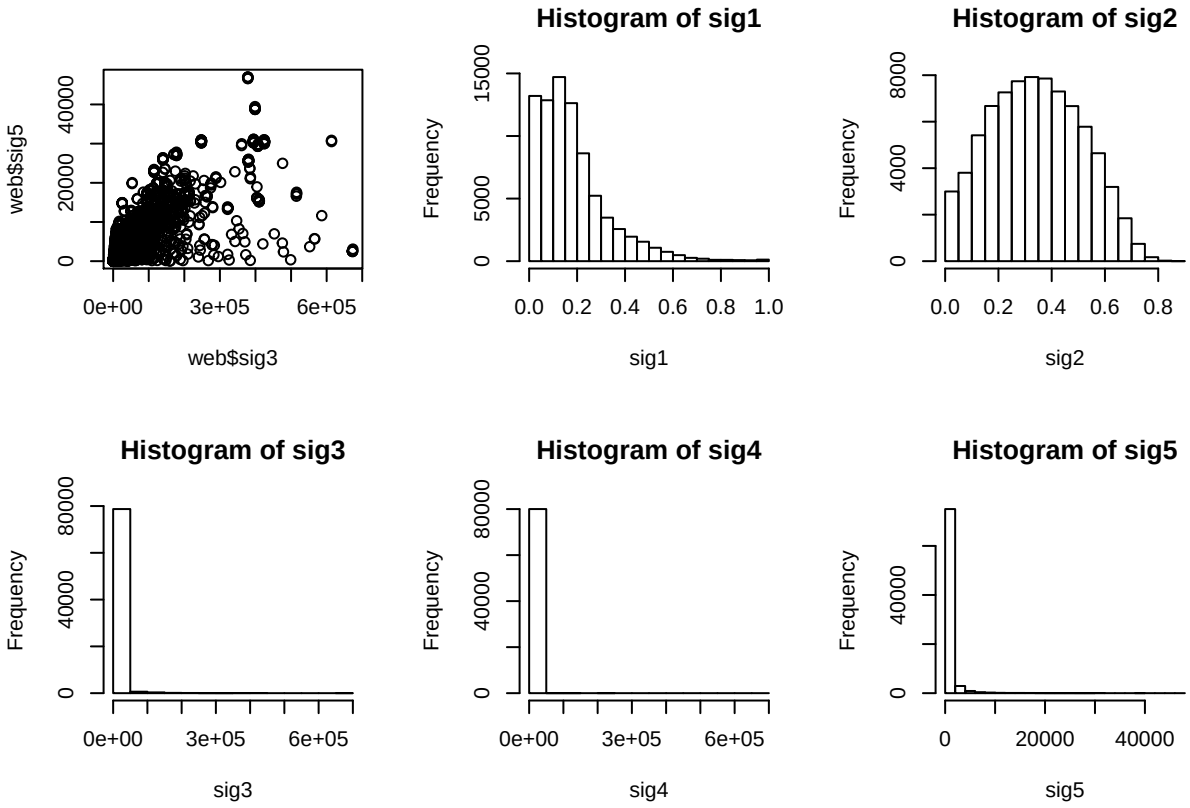
August 3, 2016

Introduction

The goal of this project is to build a binary classification model to classify the relevance of web searches according to 10 features, sig1 through sig8, *query_length*, and *is_homepage*. *is_homepage* is a binary variable encoded with 1 and 0 and the rest of the features are quantitative. Relevance is encoded as 1 - relevant and 0 - not relevant. In addition to the 10 features, *query_id* and *url_id* are included in the data set. The training data consists of 80,046 observations. Predictions are made on a test data set consisting of 30,001 observations and stored in a text file.

Selection

The dataset does not have any missing data. In the training set, 44% of observations are classified as 1-relevant and 56% of the observations are classified 0-irrelevant. Because the posterior probability of relevance is pretty evenly split, the threshold used to classify probabilities from all methods is kept at the default of 0.5. From the correlation matrix, sig3 and sig5 are highly correlated with a correlation coefficient of 0.81. To address this collinearity problem, sig3 and/or sig5 should be omitted. Histograms of all the predictors show that sig2, sig7, sig8 are normally distributed. They are also the most important predictors. Transformations were to be applied to the other predictors to make the distribution more normal for Logistic Regression and SVM.



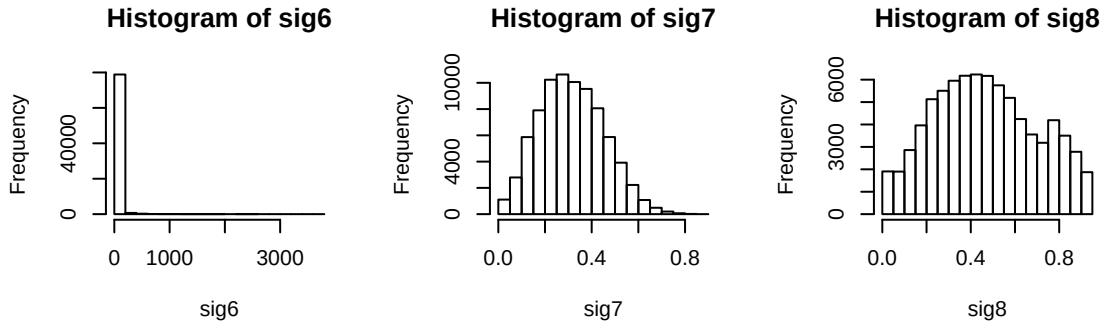


Figure 1: Predictors

Feature Selection

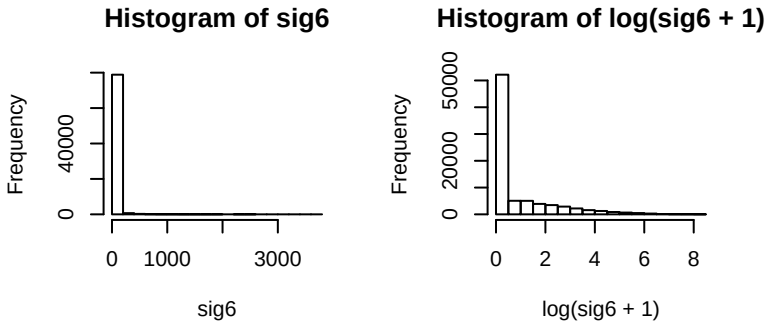
Feature selection is performed with all features present using recursive elimination in the caret package. A random forest algorithm is performed on each iteration with 10-fold cross validation to find the most important predictors. The top predictors are sig2, sig6, sig1, sig7, sig5, which matches results from the correlation matrix.

Table 1: Correlation to Relevance

Predictor	Correlation
query_id	-0.0003054716
url_id	-0.0216082025
query_length	-0.0000543074
is_homepage	0.0602881029
sig1	0.1601978917
sig2	0.3058150785
sig3	0.0727437899
sig4	0.0342550116
sig5	0.1032248568
sig6	0.1245134091
sig7	0.1651414342
sig8	0.0287305544

Transformations

Features sig1, sig2, sig7, sig8 range between 0 and 1, while sig3, sig4, sig5, and sig6 have a much larger range of values. To make the distribution of the features more normal, a log transformation was applied after adding 1 to the observations since there are many occurances of 0.



Data Mining

Data is split into a training set and a test set of equal size by taking a random sample from the entire data set. The misclassification error is calculated on the test set for Naive Bayes, Decision Tree with Random Forest and Boosting, K Nearest Neighbors, Logistic Regression, and SVM.

Decision Tree

No preprocessing of the predictors is done prior to feeding into a decision tree. Starting with all features, the rpart library generated a decision tree with 2 branches consisting of only one predictor sig2. The test error is 0.361742. Since there are only 2 branches, not much pruning can be done.

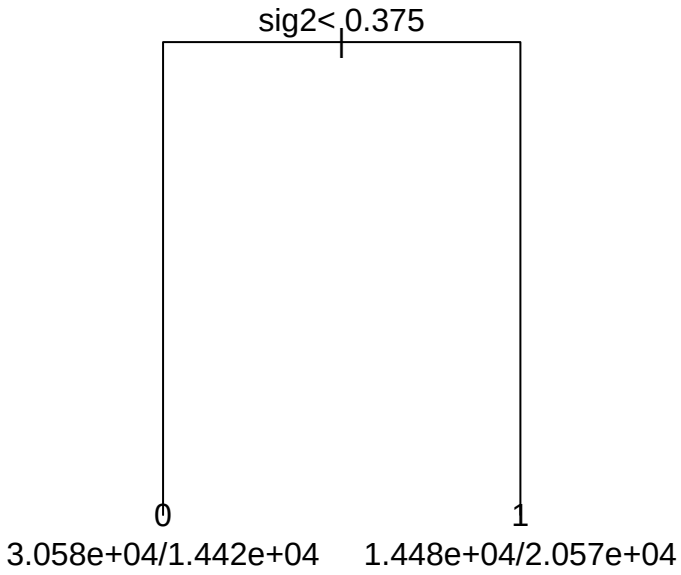


Figure 2: Decision Tree

Bagging

Bagging is implemented with the randomForest library with all predictors and the out of bag error is reported. Bagging lowered the test error to 0.3425 with mtry=10 and ntree=5000. The importance plot identifies the most significant variables with sig2 at the top.

Random Forest

Random Forest improves upon Bagging by reducing mtry. Random Forest with mtry=5 lowered to the test error to 0.3393 with ntree=5000. Futher tuning Random Forest may improve the test error but is computational expensive.

Boosting

Boosting is implemented with the gbm library as described in the textbook with a bernoulli distribution. Boosting with 5000 iterations gives test error of 0.3431277, very similar to Bagging and Random Forest methods. Plotting the ROC curve shows boosting could be a good classifer for this data set. Boosting with 5000 trees took much less time to run than Bagging and Random Forest with 5000 trees.

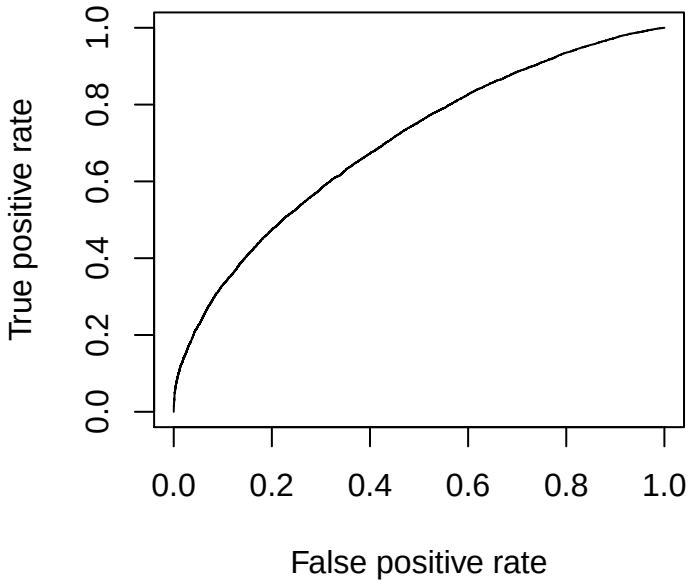


Figure 3: ROC Curve. The ROC Curve for other methods look identical to this one so they are not shown.

Table 2: No. of Trees and Test Error

Trees	Test Error
1000	0.3555955
3000	0.347775
5000	0.3431277

Naive Bayes

The Naive Bayes implementation in package e1071 was applied to the data. 10-fold cross validation performed with the caret package yielded a test error of 0.3915499 with all features. Including LaPlace smoothing had negligible effects on the test error. Using only features in the 7 variable model chosen by feature selection ($query\_length + is\_homepage + sig1 + sig2 + sig6 + sig7 + sig8$) reduced the test error to 0.3754591.

KNN

Tuning K Nearest Neighbor with 10-fold cross validation found K=3 to have the lowest test error of 0.4083899. All variables except for relevance were scaled to have zero and unit variance. Because the KNN classifier predicts the class of a given test observation by identifying the observations that are closest to it, the scale of the variables matters. Variables on a large scale will have a larger effect on the distance between the observations. Several of the variables have a range in the 10,000's so they have to be scaled.

Logistic Regression

Logistic Regression with 10-fold cross validation was applied to the data set. To get a sense of the importance of each predictor, one predictor was omitted every time and the 10-fold cross validation test error was calculated from fitting Logistic Regression on the rest of the predictors. Omitting sig2 increased the test error the most, which is consistent with other methods suggesting that sig2 is the most important variable. This method of selecting features is not as complete as best subset selection which looks at 2^p subsets of predictors but given time constraints, it is good enough to get a general idea of which predictors are the most important.

Table 3: Logistic Regression: Omitting one variable

Variable Omitted	Test Error
sig2	0.3776079
sig6	0.3533968
sig7	0.349574
sig8	0.3484247
is_homepage	0.3484746
query_length	0.3490993
sig3	0.347875
None	0.347825
sig4	0.3475502
sig5	0.3475252
sig1	0.3475002

From omitting variables in Logistic Regression, it is clear that the most important variables are sig2, sig6, sig7, sig8, *is_homepage*, and *query_length*. Using only these 6 variables reduced the test error to 0.3474752. It is unclear whether sig3 or sig1 has a greater effect so both were tried in the logistic model. The distribution of sig1 is skewed so taking the square root transformation made the distribution more bell shaped and also improved the test error.

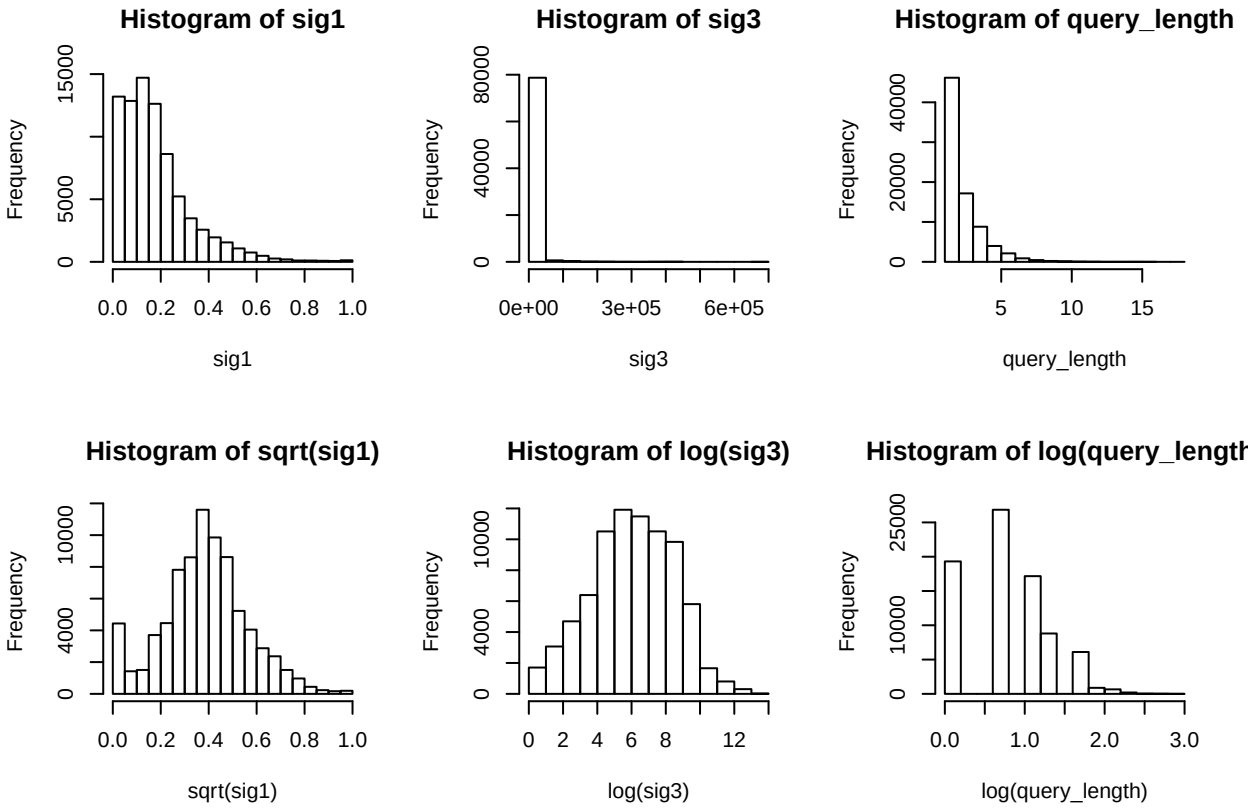


Figure 4: Transformations

Interactions of *query_length* with the other variables reduced the test error to 0.3463009. Most of the predictors have many occurances of 0 so 1 was added to all the observations to perform a log transformation on the predictors. sig6 has a mean of 14, maximum of 3645, but its 3rd quartile is only 2, suggesting that a log transformation will make the distribution more normal. A model consisting of *is_homepage* +

$query\_length * sqrt(sig1) + query\_length * sig2 + query\_length * log(sig6 + 1) + query\_length * sig7 + query\_length * sig8$ gave a test error of 0.3407041. Taking a log transformation of $query\_length$ and retaining all the interactions reduced the test error to 0.3389301. Additional log transforms reduced the test error slightly but fitting was stopped at this point to avoid further complicating the model. Final model: $is\_homepage + log(query\_length) * sqrt(sig1) + log(query\_length) * sig2 + log(query\_length) * log(sig6+1) + log(query\_length) * sig7 + log(query\_length) * sig8$ To get a measure of how good the Logistic Regression classifer is, the ROC curve is identical to Figure 4 and AUC is calculated to be 0.7042719.

SVM

SVM with linear and radial kernels were applied to the data using the 7 variable model with interactions and log transformation from Logistic Regression. $is\_homepage$ was not included because it is a binary variable. The final model is: $log(query\_length) * sqrt(sig1) + log(query\_length) * sig2 + log(query\_length) * log(sig6 + 1) + log(query\_length) * sig7 + log(query\_length) * sig8$. The SVM implementation in package e1071 scales all predictors and response variables to have unit variance and zero mean. Tuning the SVM on the entire data set for optimal cost using the tune function in package e1071 gives a misclassification error from 10 fold cross validation. Using this optimal cost, SVM was fit on the training set to generate predictions on the test set. Given the computational cost of tuning the SVM, more tuning could be done to find a more optimal cost, but it may not be able to do that much better than Logistic Regression.

To compare SVM with Logistic Regression, the ROC curve is plotted on the test set, identical to Figure 4, and the area under the curve is computed. For a linear kernel with cost = 0.1, the 10-fold cross validation error is 0.3390821 but the AUC is 0.7023767. For a radial kernel, the best cost is 0.001 with 10-fold cross validation error of 0.347104.

Conclusions

Table 4: Summary of Data Mining Methods, Test Error, AUC

Method	Test Error	AUC
Naive Bayes	0.3754591	
Logistic Regression	0.3389301	0.7042719
Decision Tree	0.361742	
Bagging	0.3425	0.6978235
Random Forest	0.3393	0.7021577
Boosting	0.3431277	0.6994102
KNN (k=3)	0.4083899	
SVM Linear	0.3390821	0.7023767
SVM Radial	0.347104	

Naive Bayes and KNN had the worst performance. Very simple models such as the Decision Tree gave very good test error with just one predictor. Tuning the decision tree with Bagging and Random Forest improved the test error to 0.34, just slighter higher than Logistic Regression. SVM and Logistic Regression have very similar test errors and AUC. Given the computational cost of tuning an SVM on a large data set, Logistic Regression is a much more efficient choice. 10-fold cross validation for Logistic Regression can be computed in a matter of seconds. In addition to the computational cost, complex methods do not necessarily perform better than simple models. I chose to submit the predictions generated by Logistic Regression because it had the lowest test error and highest AUC.