Topic Prediction Using Decision Trees on the AO Dataset

Priyadarshini Iyer

Seema Puthyapurayil

Introduction

What is Avaaj Otalo?

Avaaj Otalo (literally, "voice stoop") is an interactive voice application for small-scale farmers in Gujarat, India. It provides farmers with access relevant and timely agricultural information over the phone. This service was designed in the summer of 2008 as collaboration between UC Berkeley School of Information, Stanford HCI Group, IBM India Research Laboratory and Development Support Center (DSC), an NGO in Gujarat, India. By dialing a phone number and navigating through simple audio prompts, farmers can record, browse, and respond to agricultural questions and answers.

Problem, Insights and Solution

One of the most interesting features on this platform is push notification that goes out to all the subscribers every week. The push-content contains information about some popular themes for crops. "Push-content is developed by polling a random set of farmers each week to elicit a representative set of concerns. Treatment group respondents have on average listened to 65%, or approximately 90 minutes, of push call content."[1]
In order to increase the participation of respondents to push notifications, we thought that it would be more useful to send relevant content based on the village or the time of the year. Thus, with the dataset that we had, after a lot of brainstorming, we decided to predict the topic that the farmer would be most interested in knowing more about.

Based on the topics covered in class and some preliminary research on Google Scholar, we selected Decision Trees as the algorithm of our choice in order to predict the topic. We thought that since decision trees are like white boxes we could understand the results of the classification better. You can look into the trees, clearly understand each an every split, see the impact of that split and even compare it to alternative splits. We also thought that the decision tree algorithm was simple, and we would be comfortable handling it in the limited time frame. Furthermore, we learned that the construction of decision tree classifiers does not require any domain knowledge or parameter setting, and therefore is appropriate for exploratory knowledge discovery[2]. (Han and Kamber, p292)

Data

The data we procured from the researchers was a result of the study conducted by them on a treatment group of 800 from August 2011 to September 2012. It is primarily of the following two types:
a. The usage logs of the platform
b. The demographic information about the 800 subscribers

---

[1] Cole, S., & Fernando, N. (2012). *The Value of Advice : Evidence from Mobile Phone-Based Agricultural Extension.*

[2] Han, J., Kamber, M., & Pei, J. (2006). Data mining: concepts and techniques. Morgan kaufmann.

Solution: techniques, failures, examples

Data Preprocessing

Initially, we had call log data, which included information about the caller's user id, the time and date of session, the forum that he/she accessed, the crop they asked about etc. This data was not linked to the users' demographic data, which was provided separately.

We thought that it was important to include the demographic information in the decision tree analysis so we joined the two datasets based on the user id. For every record in the call logs, we did a VLOOKUP using Excel to find fill in the demographic details of the user who had made the call.

Once we joined the two datasets, we did a cursory analysis by visualizing the data using basic charts and we found that doing monthly predictions would make more sense than doing daily or weekly, because topics related to crop would get talked about in specific months. So we extracted only the months from the timestamp and used it as an attribute.

Converting categorical data

When we started making attempts at classifying using scikit-learn library, we realized that it doesn't not take categorical data as input. Many of our attributes were categorical so we had to vectorize the data to convert it to numeric form.

Missing values

We had some call logs with the crop name and topic rows missing. Since they did not even have the topic of the call, which was the label for our classification we decided to eliminate these rows.

Feature Selection

We had a lot of attributes that could be used as input features to the classifier. At first we tried to build the decision tree using all the attributes provided in the dataset. This created a very busy tree with very less accuracy, precision and recall. We then decided to use information gain to rank the attributes to see which attributes had insignificant information gain and eliminated them one by one. This really helped us in reducing the features and making the tree size considerably smaller. We also decided to eliminate some features like age of the farmer calling, because we felt that the researchers wouldn't be able to provide this as an input. The following table represents the values of information gain for different features after preliminary analysis on Weka.

Ranked attributes

Info gain	*Rank*	*Crop*
0.4081	1	Messagecrop
0.3463	2	Listendurations
0.125	3	Month
0.0447	4	Forumid
0.0416	5	Village

Training & Classification

"Information gain, as we saw, is biased toward multivalued attributes. Although the gain ratio adjusts for this bias, it tends to prefer unbalanced splits in which one partition is much smaller than the others. The Gini index is biased toward multivalued attributes and has difficulty when the number of classes is large. It also tends to favor tests that result in equal-sized partitions and purity in both partitions. Although biased, these measures give reasonably good results in practice." (Han & Kamber, 2006) We trained the classifier using scikit-learn library's Decision Tree Classifier setting the split criterion as "entropy" which will make the classifier judge the quality of split based on information gain. The classifier was trained using 83,500 records and set aside the remaining 200 records selected randomly for testing. We trained our classifier with the following top 3 features: Crop, Duration and Month

The decision tree that resulted from the classification can be found at "AOMining/data/decisiontree.txt" in the github repository.

Testing

From the original dataset, we picked 200 records and set them aside for testing (didn't use them for classification). Once the training and classification was done we found that the decision tree could predict the topic of the calls with an average accuracy of 66%. For additional metrics, please check "AOMining/data / OutputClassifier" on the github repository. We looked at the results and realized that it would make more sense to predict the probability of occurrence of each topic, rather than predicting just one topic. After a discussion with peers and among the team, we decided to go ahead with changing the algorithm.

The decision tree now shows an output like this:

```
priya@ubuntu:~/workspace/AODashboard/classify$ python topicPrediction.py
Enter 1 for demo and 2 for entering your data 2
Please enter as shown in the examples because there are no input validations yet!:)
Enter the crop name. Your options are 'Cotton', 'None', 'Cumin', 'Mustard', 'Other', 'Wheat', 'Castor', 'Groundnut', 'Onion', 'Sorghum
', 'Brinjal', 'Chilli', 'Gram', 'Paddy', 'Millet', 'Sesame', 'Banana', 'Maize', 'Garlic', 'Tobacco', 'Papaya'  Cotton
Enter the average call duration in seconds Eg: 54,45,55  5
Enter the month name in short. Eg: Jan, Feb, Mar  May
Input more? y/n: y
Enter the crop name. Your options are 'Cotton', 'None', 'Cumin', 'Mustard', 'Other', 'Wheat', 'Castor', 'Groundnut', 'Onion', 'Sorghum
', 'Brinjal', 'Chilli', 'Gram', 'Paddy', 'Millet', 'Sesame', 'Banana', 'Maize', 'Garlic', 'Tobacco', 'Papaya'  Cumin
Enter the average call duration in seconds Eg: 54,45,55  45
Enter the month name in short. Eg: Jan, Feb, Mar  Nov
Input more? y/n: n
Test Prediction Results
The top row is the actual record: crop,actual topic, call duration, month.
The percentage-wise breakdown of the predicted topic are below each of the actual records.
Record #0 : ['Cotton', 5, 'May']
['Crop Planning: 53.3%']
['Diseases: 6.7%']
['Fertilizers/Bio-organic: 6.7%']
['Marketing: 6.7%']
['Seeds: 26.7%']
**************************
Record #1 : ['Cumin', 45, 'Nov']
['Seeds: 100.0%']
************************
priya@ubuntu:~/workspace/AODashboard/classify$
```

Here, the user enters Cotton as the crop, the average duration as 5 seconds and the month as May, he has the option to enter more records and he enters Cumin, 45 seconds and Nov. The classifier then predicts the topic of the calls for both records. For the record [Cotton, 5s, May] the prediction is that the topic could be Crop Planning with a 53.3% probability, Seeds with a 26.7% probability and so on.

References

We also used chapter 6 from the book "Data Mining and Concepts" by Han and Kamber to understand the process of classification and prediction. It was really helpful to learn about the different split criteria and how that can affect the decision tree classifier. We also used the chapter to understand the preprocessing steps to be taken before the classification.

We used the scikit-learn examples given in the documentation, as references to create our decision tree. We started by implementing these examples to understand how the code works and then writing the code for our decision tree. Some tools that really helped us along the way include Excel for preprocessing data, Tableau and Excel for cursory evaluations of data through visualizations, and most importantly we used Weka to explore the different types of machine learning algorithms.

Future work

The decision tree could be further enhanced if there was more than a year's worth of data. We tried contacting the researchers for more data but we were unable to get more data from them.

We also think that we would have more control over the algorithm and training if we wrote the code for the classifier ourselves. We would have more control over the tree and the splitting criteria. scikit-learn also didn't allow us to prune the tree, so we had a lot of trouble exporting the extremely large tree into graphviz.

We didn't use S3 to run our classifier and currently we are running it on our own machines. Once we get more data to enrich our training set, we could run the classifier on S3 clusters for faster performance.