Kaggle Competition: what’s your favorite Xbox game ?

Our team won the 6th position among 95+ teams in this competition which ended September 2012. This posting will describe the algorithms and techniques employed in our approach.

**Data Description**

The training set included information about user interest in Xbox games. One line of the training set indicates the user id, the actual search query, the product the user clicked on as a result of the query, the query timestamp and the click timestamp. The test set had all information except the game favored by the user, which was to be predicted. 5 predictions had to be made and a Mean Averaged Precision (MAP) metric was used to evaluate the prediction.

**Important trends and observations**

Notable features included the strong influence of time in predicting user behavior. This could be because a certain game or its version was released recently or had news value. To include the influence of time, the time period for which the user information was available (roughly 4 months) was divided into 12 time zones. The Naïve Bayes training parameters (described later) were learned separately for each time zone. To reduce noise, smoothing was performed on the data (local average) w.r.t time.

**Feature Engineering**

Feature Engineering is undoubtedly the most important piece in the puzzle. The winning teams in most Kaggle competitions use wildly different approaches – SVMs, kNN, randomForest etc. However, the common factor is usually the care with which feature set is extracted and cleansed. A package such as libSVM for SVM will guarantee performance for any user, so the key differentiation is the feature set extracted and provided to the learning algorithm.

* We used unigrams; essentially each word occurring once in the query set for all users is a feature.
* Bigrams. Naïve Bayes places strong assumptions on independence of words. In practice, xbox and 360 would be highly correlated in the Xbox game context. To include this information, we used bigrams as features. For “gears of war xbox”, the features would be “gears war”, “war xbox” and “gears xbox”. Notice that the words in the bigram are in sorted order.

**Natural Language Processing**

We used spelling correction and lemmatization in our approach. WordNet was used for this purpose. The corpus made available in the data provided as well as some generic corpuses were used to collect normal words. These were lemmatized.

The user queries had several spelling errors. Also, many words could be associated together as they had the same meaning. The following approaches were utilized:

* Edit distance with words in the corpus (after lemmatization) to correct spell errors.
* Lemmatization grouped words with the same sense in one group
* Numbers and words were separated – eg “xbox360” as “xbox” and “360”. This would provide better features for the classifier.

**Core Algorithm and Combining Classifiers**

We used a Naïve Bayes based learner as the core algorithm. Two different learners were trained – one focused on unigrams, whereas the second focused on bigrams. A boosting approach was used to combine the results. Boosting is the process where the residue from a weak classifier is provided as input to the second classifier, which would naturally focus on areas with higher modeling error. Due to time constraints, we used a simple linear combination of probabilities obtained from the models. The linear parameters used to combine the models were obtained using cross-validation.

**Conclusion**

The competition highlighted the need for collaboration in data mining and scaling. The most important lesson learned is the need to do early visualization of data and feature engineering. This saves a lot of time and effort, and avoids premature optimization. Even simple learning algorithms could give good results if the feature set is intelligently extracted.