2016 - Home Depot Product Search Relevance Kaggle Competition

D'Souza, Ajay
ajaydsouza@gatech.edu

Abstract

The goal of this Kaggle Competition is to develop an algorithm that matches the results of the manual rating score for determining the relevance of a search string to a given product

Contents

1 Project Description

- `www.kaggle.com` has a an open competition for **Home Depot Product Search Relevance** at `https://www.kaggle.com/c/home-depot-product-search-relevance`. At Home Depot humans rate the relevancy of a search string to a product by assigning an rating score to a tuple of $\langle product, searchstring \rangle$. The on-line search algorithm at Home Depot is evaluated and tuned to use this manual rating score. Manual rating however is a slow process and will not scale. Through this competition Home Depot is seeking to find an algorithm than can learn from the manual rating process and mimic it.

- Home depot has provided a training data set that has a tuple of $\langle product, searchstring \rangle$ along with a rating score for each tuple. The rating score is in the range of $1 \ldots 3$, with 1 indicating that the product is irrelevant to the search string and 3 indicating that the search string is fully relevant to the product. We need to train a model to be able to predict this rating score for a tuple of $\langle product, searchstring \rangle$.

- A test data set comprising of tuples of $\langle product, searchstring \rangle$ is provided. The model needs to make predictions for this test set. The test results set are to be submitted and will be evaluated using RMSE (Root Mean Square Error).

- It should be noted that the goal is not to search for the best product for a search string, but instead to develop a model which predicts a rating score that matches the manual rating score for a tuple of $\langle product, searchstring \rangle$.

2 Source of Data

- The data for the competition is provided on Kaggle at `https://www.kaggle.com/c/home-depot-product-search-relevance/data`. The data comprises of training data, test data, a catalog of product descriptions, product attributes, instructions for manual rating. The table (1) lists the data files provided along with a description as mentioned on `https://www.kaggle.com`

train.csv	The training set, contains products, searches, and relevance scores
test.csv	The test set, contains products and searches. You must predict the relevance for these pairs
product_descriptions.csv	Contains a text description of each product. You may join this table to the training or test set via the product_uid
attributes.csv	Provides extended information about a subset of the products (typically representing detailed technical specifications). Not every product will have attributes.
sample_submission.csv	A file showing the correct submission format
relevance_instructions.docx	The instructions provided to human raters

Table 1: Data for - **Home Depot Product Search Relevance** Competition

- Table (2) provides a detailed description of the contents of the data provided in the training dataset, product description and product attributes files.

id	A unique Id field which represents a (search_term, product_uid) pair
product_uid	An id for the products
product_title	The product title
product_description	The text description of the product (may contain HTML content)
search_term	The search query
relevance	The average of the relevance ratings for a given id
name	An attribute name
value	The attribute's value

Table 2: Data description - **Home Depot Product Search Relevance** Competition

3 Scientific Questions to be Explored

- The predictor data provided is essentially bunch of text. Hence this cannot be fit directly to any data mining or statistical learning processes. The data needs to be preprocessed

- The success of the project hinges on using this predictor lexical content to project the tuple of $\langle product, searchstring\rangle$ into a vector space that is a meaningful representation of this tuple.

- To achieve this objective we need to engineer a feature vector for each tuple based on its product description, product title, product attributes and the search search string in the tuple

- Since all the content is lexical, engineering the feature vector involves experimenting with one or more of the following NLP techniques in addition to the rules provided for manual rating

    1. Bag of Words
    2. TFIDF
    3. Word2vec and text2Vec
    4. NLP tokenizing, Stemming , Lemmatizing, n-gram, feature and sentiment extraction

- With a feature vector in place for each tuple of $\langle product, searchstring\rangle$, we can then experiment with various machine learning techniques to train and evaluate models for predicting a rating score for any tuple of $\langle product, searchstring\rangle$
- So essentially this project has three steps
    1. Come up with a methodology to generate a intelligent feature vector for each tuple of $\langle product, searchstring\rangle$. With this we have the tuple of $\langle product, searchstring\rangle$ projected into a meaningful vector space.
    2. Train various machine learning models on these engineered features using cross validation. Evaluate the trained models by bootstrapping to choose the best model based on MSE.
    3. Use the best model to generate the test results for submission
- The following section outlines the specific models which could be considered for the purpose

4 Proposed Statistical Methods

4.1 Engineer Feature Vector for $\langle product, searchstring\rangle$

The feature vector for a given a tuple of $\langle product, searchstring\rangle$ will be generated using

1. The rules provided for manual rating
2. A combination of one or more of lexical techniques of NLP tokenizing, Stemming , Lemmatizing, n-gram, feature and sentiment extraction, Bag of words, TFIDF, text2Vec and Word2Vec.

Since the efficacy of the feature vector would depend on the model. The methodology for generating the feature vector for a given model will also be chosen by cross validation.

4.2 SVD to remove noise

Singular Value Decomposition of this matrix will filter noise and maximize the variance as follows

- We form a matrix of the feature vectors of all the tuples of $\langle product, searchstring\rangle$ in the training data set

- We perform a Singular Value Decomposition on this matrix
- We pick the higher rank terms so as to clear the matrix of noise and reform the matrix.
- This is the new search matrix, which is a projection into a new lower dimensional vector space with noise filtered
- Models will be fitted on this data using cross validation

4.3 Cross Validation and Bootstrapping

1. For the models used, the model parameters will be chosen using ten fold cross validation on the training set
2. The performance of the optimum model will be evaluated using bootstrapping on the training set. A $50 : 50$ split will be used for training and testing.
3. A statistical test (T-test, Wilcox test) on the bootstrapping results, will be used to choose the best performing model
4. The best performing model is then retrained using the whole training set
5. This model will be used to generate the test results for submission

4.4 Prediction Models

The following prediction models will be considered

4.4.1 Boosting - Random Forests

- The rating score depends largely on the interaction between the various features engineered for tuples of $\langle product, searchstring \rangle$, rather than on the independent presence or absence of particular features in the feature vector
- Decision trees by design handle the interaction in the predictors. So they are very well suited for a dataset like this in comparison to regression models where we have to specifically add the interaction terms
- Thus we can train a decision tree based on the feature vectors engineered for the tuples of $\langle product, searchstring \rangle$ in the training data and then use it to make predictions on the test data

- We will experiment with using random forests and boosting to train the decision tree models for this training set

4.4.2 Regression Models

As discussed above regression models need to have explicit interaction terms defined in order to capture the interaction between predictors in the feature vector. So for a dataset like this ,where the results depend on the interaction of predictors, it is hard for regression models to outperform a specialized decision tree model like random forests and boosting

Logistic Regression

- The distribution and variance is not known for this dataset. Logistic regression does not assume a distribution and variance.
- The rating score which is the response for this dataset is range bound between $1 \dots 3$. logistic regression ultimately predicts a probability which is in the range bound between $0 \dots 1$
- So we can try to fit a logistic regression model to this dataset as follows
- To begin we convert the rating score provided in the training set into an odds ratio as $\frac{rating}{3-rating}$
- We will then try to fit a logistic regression model to this training data using the engineered feature vector for tuples of $\langle product, searchstring \rangle$ in the training data as predictors
- For predictions, the odds output by trained logistic regression model can then be scaled this back to the rating score in range of $1 \dots 3$

4.4.3 Other Models

In addition to the two models mentioned above we will also try generalized additive models, Support Vector Machine and ensemble methods of stacking and boosting for combining various models. The details of implementation are open and will be decided during execution go the project.