**Data Analytics Term Project**

Group: Katherine Rodgers and John Merranko

**Introduction**

The purpose of this project is to predict the relevance of search results on HomeDepot.com (as compiled for the 1/18/2016 Kaggle competition) through data processing and advanced text mining techniques. The data consist of search terms (strings of text) and products resulting from each search. Product data include product names, descriptions, and for some products, a collection of attributes (all strings of text). In the training dataset, raters have rated each pair of searches and resulting products on a three-level relevance scale (1 = irrelevant, 2 = partially or somewhat relevant, 3 = perfect match), and the task is to train a model using these data to try and predict the relevance of search and resulting product pairs in the test dataset. Model performance will be evaluated on the Root Mean Squared Error (RMSE) defined below:

**Data Processing**

We have implemented the initial data processing using Python. The data within the provided product\_descriptions.csv file consist of a product\_uid and description for each product. The train.csv file includes an id, product\_uid, product\_title, search\_term, and relevance value for each search. The attributes.csv file contains differing amounts of additional information for a subset of the products and consists of a product\_uid, name, and value in each line of text. The name refers to an attribute’s name and the value is the value for that specific attribute name. The first part of processing the data was to convert the comma separated files into a more easily separated format for Python due to the many commas within the product description and attribute data. This was done using R to initially read in the csv files and then write them using a pipe separated text file format. The Python script reads in the pipe separated train, attributes, and product description files where each line undergoes a process of string manipulation. We utilize the Natural Language ToolKit, NLTK, to remove non-essential stopwords for each line. We also remove some punctuation within each line of text to get only a series of words. Due to some of the typos within the provided data files and also the process of removing punctuation, issues of camel-case scenarios and letter-digit combinations are also accounted for. We also converted all strings of text to lowercase. Due to the fact that any given product could have any number of attributes and the names of attributes are non-uniform among the products, product data can be considered highly unstructured. Because of this, we decided to simplify the data for each product by collapsing the non-brand-name data in one concatenation per product. We kept brand separate from the rest of the data due to its higher relevance as a predictor. For each product, the Python code iterated through all the attributes associated with that product and, aside from “MFG Brand Name”, concatenated each to the product\_description string which we then named product\_details. We arranged the data in JSON format with product\_brand, product\_details, and product\_uid as keys. The following is an example of a product in JSON format:

{

"product\_brand": "simpson strong-tie",

"product\_details": "not angles make joints stronger also provide consistent straight corners simpson strong-tie offers wide variety angles various sizes thicknesses handle light-duty jobs projects structural connection needed some bent skewed match project for outdoor projects moisture present use zmax zinc-coated connectors provide extra resistance corrosion look z end model number versatile connector various 90 connections home repair projects stronger angled nailing screw fastening alone help ensure joints consistently straight strong dimensions 3 in x 3 in x 1-1/2 in made 12-gauge steel galvanized extra corrosion resistance install 10 d common nails #9 x 1-1/2 in strong-drive sd screws install 10 d common nails #9 x 1-1/2 in strong-drive sd screws galvanized extra corrosion resistance made 12-gauge steel dimensions 3 in x 3 in x 1-1/2 in help ensure joints consistently straight strong stronger angled nailing screw fastening alone versatile connector various 90\u00b0 connections home repair projects product width in 3 product weight lb 0.26 product height in 3 product depth in 1.5 number pieces 1 material galvanized steel gauge 12",

"product\_uid": 100001

}

We output all JSON-formatted products to a text file with each product on a separate line. We then accessed the data file in R via the fromJSON fuction in the jsonlite library. In R, we also converted the training and testing data to lowercase. We then separated the numbers from the data before using the bag\_o\_words function from the qdap library because the function would ignore them. After making the word bags, we added the numbers to their respective bags. We then used these bags for our modeling.

**Methods for Modeling Data**

Our design to tackle this problem utilizes a methodology similar to the term frequency-inverse document frequency (tf-idf) calculation, which seeks to model how important a word is to a document in a collection. In our design, we calculate the proportion of words or numbers in the product brand name, title, and combined description and attributes (where applicable) that matched text in the search term and use these proportions as predictors in machine learning regression models in order to predict relevance between search term and product pairs. Theoretically, if a large proportion of words or numbers in the product’s brand name, title, and/or description/attributes match text in the search term, the search result should be quite relevant. Conversely, if a small proportion of words or numbers in the product’s brand name, title, and/or description/attributes match words in the search term, the search result should be quite irrelevant.

We trained several machine learning regression models including random forest, support vector machine regression, and generalized boosted regression as well as OLS linear regression with predictors that measure the text-match proportions described above and tested each model on the provided test sample. Below are basic mathematical expressions of our implemented methods:

, *i*=1,…,n; where:

= predicted relevance score

: represents machine learning regression technique (e.g., random forest, support vector machine regression, etc.) found to have lowest RMSE when predicting as a function of predictors , defined as:

*=*

*=*

*=*

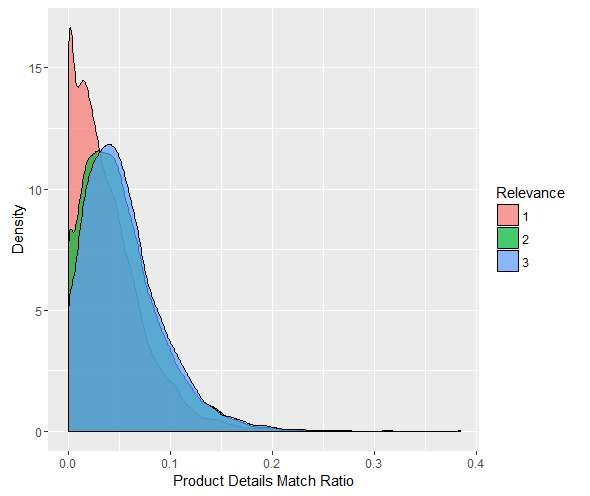
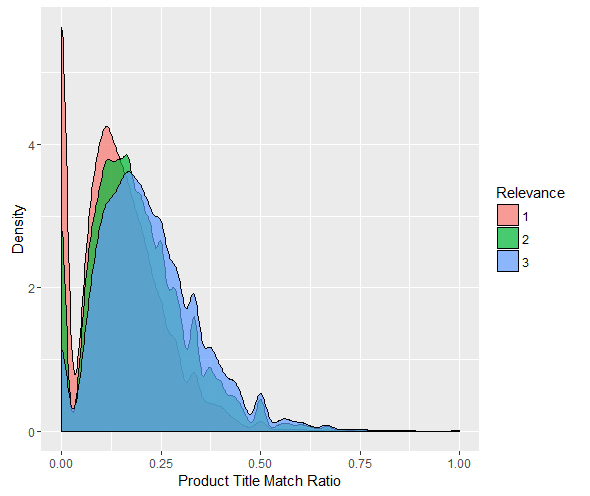
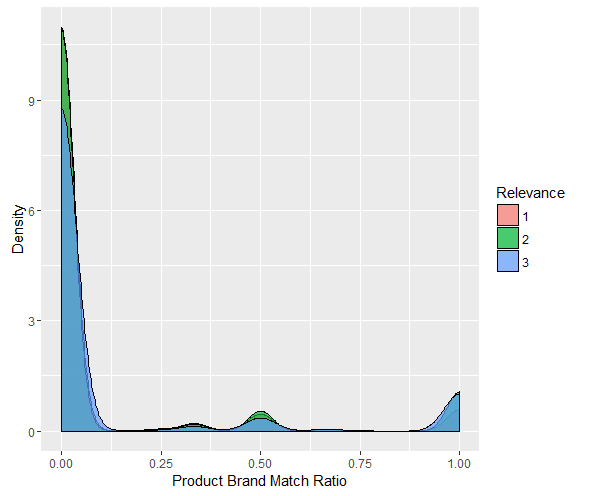
Note again that before computing predictor variables, nuisances such as stopwords, (e.g., of, and, the, etc.) and punctuation (with the exception of punctuation joining compound words or numeric expressions) were removed using bag-of-words and other string processing operations, and strings were cleaned to remedy issues in the data such as the unintended splicing together of words. The predictors were then used to train a model to predict relevance between search term and product pairs. Other parameterizations of these variables were also attempted including log transformations and dichotomizations (one or more matches vs. zero matches), but these failed to outperform the original parameterizations. As previously mentioned, we evaluate model training and testing error using the RMSE.

**Results**

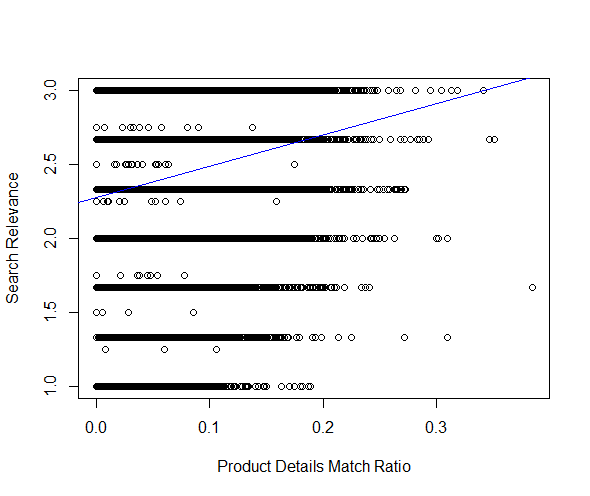
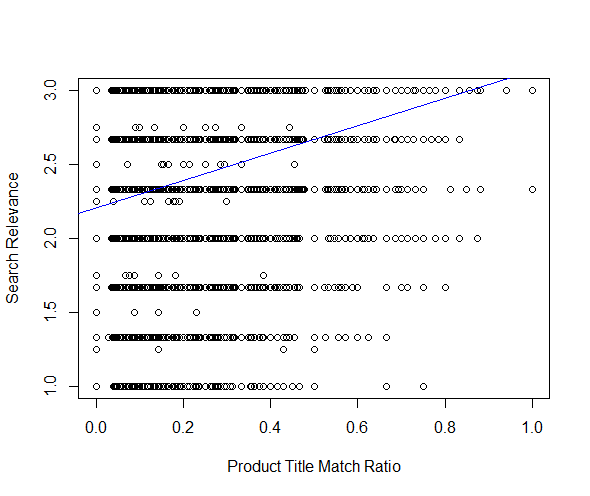
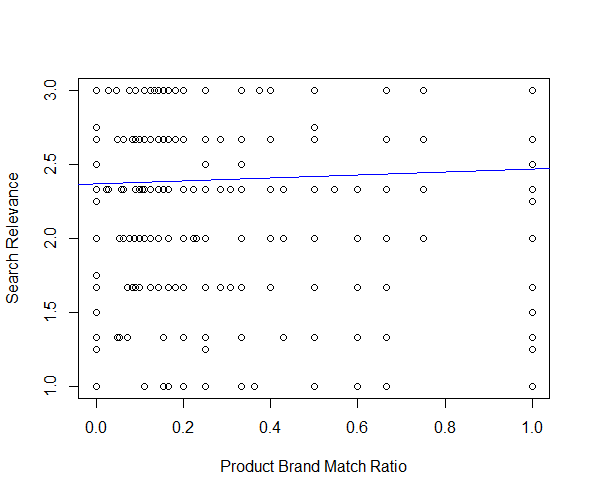
Before presenting model results, we present some graphics to assess the distributions of our predictor variables (match ratios for product brand, title, and details) and the outcome (search relevance) as well as assess the relationships between the predictors and outcome. First, below is a grid of histograms of the predictors and outcome. Search relevance is fairly left-skewed, with the majority of scores above 2. Product brand match ratio is mostly zeroes, product title match ratio is fairly normal with a slight right-skew, and product details match ratio has a smooth right-skew. Overall, most of the predictor match ratios are on the low end of the spectrum given that the majority of product information text did not match search terms.

|  |
| --- |
|  |

Next we present kernel density plots of each predictor conditional on rated search relevance score. Ideally, the greater the search relevance, the farther right the distribution of the curve should be. Overall, higher rated search relevance tended to associate with higher values in predictor match ratios.



As a final graphical assessment, we present scatter plots of rated search relevance versus each predictor match ratio along with line of best fit. While the clouds of data points are somewhat difficult to interpret, the positive slopes of each fitted line indicate a positive linear association between each predictor match ratio and rated search relevance.



To preliminarily assess the strength of our predictors, we fit linear regressions of training set search relevance on each predictor first individually and then together in one multiple regression model. The results below indicate that all three predictors have a highly significant positive association with search relevance, with the product title and details match ratios having the strongest associations.

|  |  |  |  |
| --- | --- | --- | --- |
| **Predictor** | **Coefficient** | **t-statistic** | **p-value** |
| Product Brand Match Ratio | 0.10 | 14.84 | <2e-16 |
| Product Title Match Ratio | 0.92 | 59.14 | <2e-16 |
| Product Details Match Ratio | 2.12 | 41.43 | <2e-16 |
| Product Brand Match Ratio | 0.84 | 14.25 | <2e-16 |
| Product Title Match Ratio | 0.05 | 7.34 | 2e-13 |
| Product Details Match Ratio | 0.77 | 41.57 | <2e-16 |

Next we trained machine learning models and calculated RMSEs using 50/50 split internal train/test validation. The techniques we implemented included OLS linear regression, random forest (both 500 and 1000 trees), support vector machines regression (using linear, 2nd order polynomial, 3rd order polynomial, and radial kernels), and generalized boosted regression (500 and 1000 trees). The results below indicate random forest techniques performed the best based on both the training and testing RMSE.

|  |  |  |  |
| --- | --- | --- | --- |
| **Algorithm** | **Model Details** | **Train RMSE** | **Test RMSE** |
| OLS Regression | Linear | 0.520 | 0.521 |
| Random Forest | Number of Trees = 500 | 0.515 | 0.517 |
| Number of Trees = 1000 | 0.515 | 0.517 |
| Support Vector Machines Regression | Linear | 0.522 | 0.524 |
| 2nd Order Polynomial | 0.531 | --- |
| 3rd Order Polynomial | 0.532 | --- |
| Radial | 0.520 | 0.522 |
| Generalized Boosted Regression | Number of Trees = 500 | 0.530 | --- |
| Number of Trees = 1000 | 0.528 | 0.525 |

**Conclusion**

All three of our theorized predictor match ratios were significant positive predictors of Home Depot search relevance. While further transformations of these data were attempted, none predicted search relevance as effectively as using the raw predictor match ratios themselves. As compared to OLS, support vector machines, and generalized boosted regression techniques, random forest regression predicted with the lowest training and testing error.

**Software Usage**

We used Python for the majority of the product data processing and manipulation. The product data was output to a text file and read into R. We then performed our model building, training, and testing in R.

**Leaderboard Screenshot**

