BUSINESS USE CASE FOR NLP

A tutorial on predicting the next trend in a fashion e-Commerce context

22-APR-2018

Abstract

This tutorial explains how NLP techniques can be used for predicting a product's success potential, based on the response the product receives in the form of text reviews.

Vinay Arun

vinayarun@gmail.com

https://in.linkedin.com/in/vinayarun

Business use case for NLP

A tutorial on predicting the next trend in a fashion e-Commerce context

Prerequisites

Experience with the specific topic: Novice

Professional experience: No industry experience

Knowledge of machine learning is not required, it would help if the reader is familiar with basic data analysis. All concepts in the article are explained in detail from scratch for beginners. To follow along, download the sample dataset here. This tutorial explains a business application of Natural Language Processing for actionable insights.

Introduction to Natural Language Processing and Business use cases

A very simple definition that is commonly found online for Natural Language Processing (NLP) is that it is the application of computation techniques on language used in the natural form, written text or speech, to analyse and derive certain insights from it. These insights could vary quite widely, from understanding the sentiment in a piece of text to identifying the primary subject in a sentence.

The complication of the analysis can also range from simple to very complex. For instance you may want to categorise pieces of text, say from Twitter data, as having positive sentiment or negative sentiment. On the other hand you may want to extract product suggestions/complaints from product review data from a large dataset to figure out new product launch strategies. The work being done on chatbots, to understand text queries and respond to them; spam detection, etc. falls in the ambit of NLP.

This field can encompass many applications and thus many techniques, which makes it a fairly large topic. For this particular tutorial we will be discussing in detail a business case application of NLP, to derive practical and actionable insights in the e-commerce domain.

Understanding the Business scenario

For this tutorial a dataset available on Kaggle, from the e-commerce fashion domain, is used. This is available at: https://www.kaggle.com/nicapotato/womens-ecommerce-clothing-reviews

kaggle	*This is a Women's Clothing E-Commerce dataset revolving around the reviews written by customers. Its nine supportive features offer a great environment to parse out the text through its multiple dimensions. Because this is real commercial data, it has been anonymized, and references to the company in the review text and body have been replaced with "retailer".*

This dataset seems fairly typical of most e commerce marketplaces, and so it is a good representation. It is a dataset that has numerical rating data as well as text reviews of several items of clothing for women. The items have a "Clothing ID" which is a unique identifier. There are two kinds of ratings - a numerical 1 to 5 for product rating and another rating of 1 or 0 for recommend the product or not, a binomial attribute i.e it has only two possible values. There are also other attributes that describe the category of the item, the age of the reviewer etc. If you think about it, these attributes provided in

this dataset are very typical of most such online shopping portals. In fact, this is not restricted only to the fashion domain and so the techniques applied here should be extendable to other domains as well.

With this dataset, it would be interesting to explore if the text reviews can predict whether the item would trend or not. That would be pretty useful information for a fashion business. If such insight was accurately available about a product the product or category manager can make sound decisions on the product's inventory strategy, promotions etc. This is our business problem.

So the aim is to have an automated system, that can analyse the initial few reviews by customers of a new product and predict if the newly launched product will succeed or not. With this information the products inventory can be built up and it can be pushed through the supply chain. This way one can implement soft product launches and respond to the market's acceptance of the product.

The two tools used for this experiment are Excel and Rapidminer. There are two parts to this workflow. First some feature engineering, carried out in excel, then the actual model training and testing, carried out in Rapidminer.

Part 1: Feature Engineering in Excel

With the data given, we need to first arrive at a mathematical definition for "trending product". At this stage is where domain knowledge of the business comes in handy. We know that if a product is doing well and if customers are happy with it, they take the effort to write positive reviews about it. People generally tend to write detailed reviews either if they are very happy or very dissatisfied with the item. If they are happy, they tend to write good reviews and rate it highly. Also, products that do well get many reviews, and most reviewers would have a similar opinion of the product with high ratings.

To translate this mathematically; the product's success has a direct relation between number of reviews, average rating of the product, and has an inverse relation with the standard deviation of the ratings for the product. That is if the market likes it, a lot of customers tend to rate it highly without much variation. We can create a new feature based on this understanding.

$$\begin{aligned} Product's success &\propto Number\ of\ reviews \\ &\propto Average\ rating \\ &\propto \frac{1}{Std\ deviation\ of\ ratings} \end{aligned}$$

Borrowing from the tanh activation function used in neural network training, we can define a new feature "Rating Strength" as below.

$$Rating\ Strength = (Rating\ Factor)^2$$

Where,

$$Rating\ Factor = \frac{1 - e^{\left(\frac{Reviews\ proportion * Average\ rating}{Std\ deviation\ of\ rating}\right)}}{1 + e^{\left(\frac{Reviews\ proportion * Average\ rating}{Std\ deviation\ of\ rating}\right)}}$$

Where, "Reviews proportion" is the proportion of reviews that item received from the full dataset Average and std deviations are self-explanatory.

This creates a new feature with values between 0 and 1. Values closer to 1 get assigned to items where there have been a large number of reviews, the ratings have been good and there is very low variance in the ratings, i.e everyone has rated it well. Values closer to 0 are assigned for items which did not generate much excitement – the number of reviews are few and ratings are low with high variance. The main aim of this mathematical definition was to get a numerical feature that represents the product's success and preferably with values between 0 and 1, rather than unscaled large values.

With this you can create a binomial label called "Item_success", with two possible values "Trending" or "Not trending", where "Trending" is assigned to items with Rating Strength above a certain cutoff and "Not trending" is assigned to items below this cutoff value. For this dataset I picked a cutoff value of 0.00036. This cutoff was determined by simply looking at the dataset, however there could be better automated ways of doing this. This manual decision, however, was a very simple and quick solution. For a person working with these products in question it would be a fairly straight forward decision. Below is a snippet of the dataset.

Clothing	A	Title	Review Text	Rati	Recom	Positive	Num_rev	Avg_ratii	Std_dev_ratin	Num_rec	Percent_re	Percent_nu	Rating_stre	Item_success
1077	60	Some major design flaws	I had such high hopes for this dress and really wanted it to work for me. i initially ordered	3	0	0	251	4.04	1.148	199	0.793	0.013	0.00051	Trending
1049	50	My favorite buy!	I love, love, love this jumpsuit. it's fun, flirty, and fabulous! every time i wear it, i get nothi	5	1	0	25	4.44	0.804	23	0.920	0.001	0.00001	Not trending
847	47	Flattering shirt	This shirt is very flattering to all due to the adjustable front tie. it is the perfect length to w	5	1	6	4	4.00	1.225	3	0.750	0.000	0.00000	Not trending
1080	49	Not for the very petite	I love tracy reese dresses, but this one is not for the very petite. i am just under 5 feet tall	2	0	4	241	4.26	1.063	200	0.830	0.012	0.00060	Trending
858	39	Cagrcoal shimmer fun	I aded this in my basket at hte last mintue to see what it would look like in person. (store p	5	1	1	18	3.83	0.957	11	0.611	0.001	0.00000	Not trending
858	39	Shimmer, surprisingly goes with	I ordered this in carbon for store pick up, and had a ton of stuff (as always) to try on and u	4	1	4	18	3.83	0.957	11	0.611	0.001	0.00000	Not trending
1077	24	Flattering	I love this dress. i usually get an xs but it runs a little snug in bust so i ordered up a size. ver	5	1	0	251	4.04	1.148	199	0.793	0.013	0.00051	Trending
1077	34	Such a fun dress!	I'm 5'5" and 125 lbs. i ordered the s petite to make sure the length wasn't too long. i typic	5	1	0	251	4.04	1.148	199	0.793	0.013	0.00051	Trending
1077	53	Dress looks like it's made of ch	Dress runs small esp where the zipper area runs. i ordered the sp which typically fits me ar	3	0	14	251	4.04	1.148	199	0.793	0.013	0.00051	Trending
1095	53	Perfect!!!	More and more i find myself reliant on the reviews written by savvy shoppers before me a	5	1	2	287	4.03	1.140	225	0.784	0.015	0.00067	Trending
767	44	Runs big	Bought the black xs to go under the larkspur midi dress because they didn't bother lining the skirt portion (grrrrrrrrrrr). my stats are 34a-28/29-36 and the xs fit very smoothly around the chest and was flowy around my lower half, so i would say it's running big.	5	1	0	1	5.00	0.000	1	1.000	0.000	0.00000	Not trending
1077	50	Pretty party dress with some is	This is a nice choice for holiday gatherings. i like that the length grazes the knee so it is cor	3	1	1	251	4.04	1.148	199	0.793	0.013	0.00051	Trending
1065	47	Nice, but not for my body	I took these out of the package and wanted them to fit so badly, but i could tell before i p	4	1	3	16	3.88	0.696	13	0.813	0.001	0.00001	Not trending
1065	34	You need to be at least average	Material and color is nice. the leg opening is very large. i am 5'1 (100#) and the length hit	3	1	2	16	3.88	0.696	13	0.813	0.001	0.00001	Not trending
853	41	Looks great with white pants	Took a chance on this blouse and so glad i did. i wasn't crazy about how the blouse is phot	5	1	0	6	3.67	0.943	4	0.667	0.000	0.00000	Not trending
1120	32	Super cute and cozy	A flattering, super cozy coat. will work well for cold, dry days and will look good with jean	5	1	0	2	4.50	0.500	2	1.000	0.000	0.00000	Not trending
1077	47	Stylish and comfortable	I love the look and feel of this tulle dress. i was looking for something different, but not ov	5	1	0	251	4.04	1.148	199	0.793	0.013	0.00051	Trending

Figure 1 Feature engineered dataset (snippet)

This creation of a new feature in the dataset from available features is called feature engineering. It's a useful step that can be deployed to machine learning problems. All of the activity till here was carried out in Microsoft Excel. The next part is the development of a model to predict this binomial label that we created. The predictive modelling part was carried out in Rapidminer.

Part 2: Predictive Modelling in Rapidminer

Setup

This part was carried out in Rapidminer. The problem is basically of language processing. Given a new review we want to make a prediction if that item would trend or not. There is certainly information contained in the language of a review that can give clues on whether the item will succeed. Given past data, we should be able to predict for new items being reviewed.

Essentially, we need to use past data to build a model that can learn from the language of reviews for known successful products, and use this learning on new reviews of future products to predict its chances of success.

The process flow used in Rapidminer is shown below:

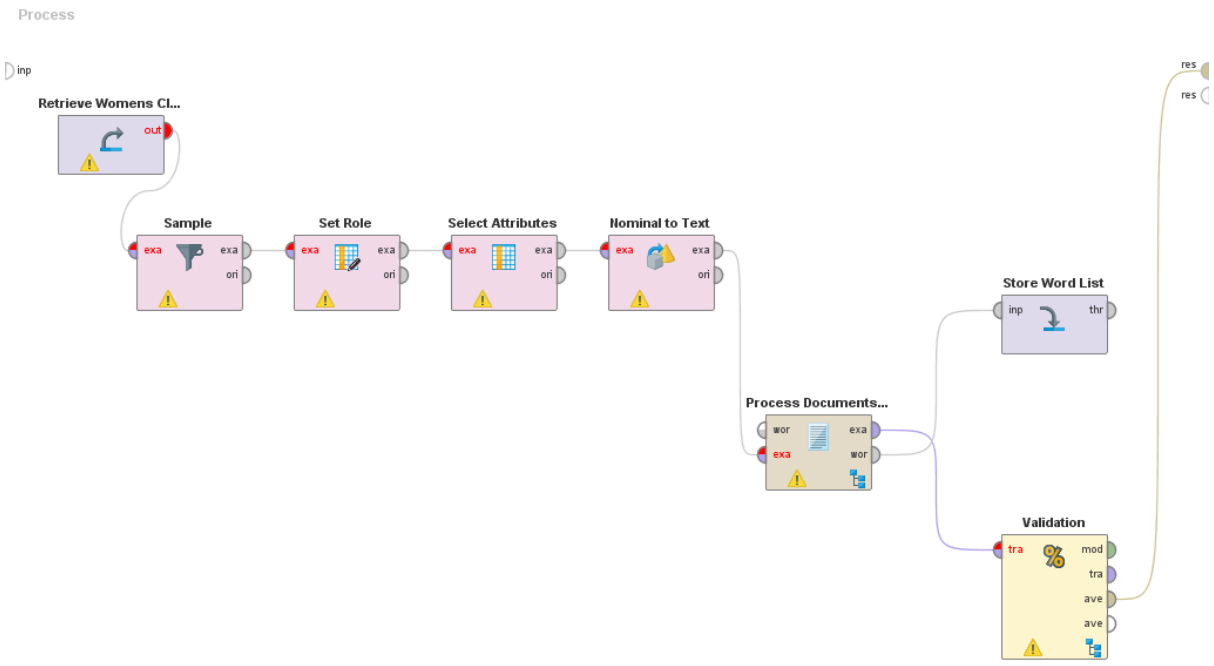

Figure 2 Model training in Rapidminer

Rapidminer is a very easy to use machine learning platform and is ideal for business managers who may not be data scientists but are keen on quick application of these techniques to their business requirements. In Rapidminer each operation can simply be carried out by dragging and dropping the related operator into the workspace and connecting the input data graphically. You can visualise the data flowing and getting processed in each stage in the above diagram.

Let us go through each of the operators used above:

1. **Retrieve**: We first need to retrieve the dataset that we created in part 1. For this either we can read the excel file itself or first import the data into Rapidminer and then access it. In this case we are doing the latter. The data will have to be imported first, which is pretty straightforward and there is a wizard to do the import step by step, you can start the import by clicking on the "Add Data" button in Rapidminer. In the Retrieve operator's settings we then define the name of the dataset, defined during import, to be accessed.
2. **Sample**: The dataset contains close to 19,500 rows. But we don't want to use the full dataset. It will be time consuming and could lead to over fitting. So in the sample operator we can define how many samples to select. I have chosen 4500 samples.
3. **Set Role**: We need to define which of the features is of interest to us, the one we want to predict. For us it is the "Item_success" column, we define this feature as the *label*. The label is the feature that needs to be predicted.
4. **Select Attributes**: In our dataset we had defined new columns like average rating, standard deviation of rating, proportion of reviews etc. to arrive at the "Item_success" feature. It is very important to remove these features from the training data that will be supplied to train the model. This is very important. These columns will not be available for future real time data and including them here could give false accuracy levels and a model that will not function in deployment. You can go through the excel file to understand these columns

better. So the features actually selected are as below:

Age
Class Name
Department Name
Division Name
Item_success
Rating
Recommended IND
Review Text
Title

All of these features will be available for new data during deployment.

5. **Nominal to Text**: This is a data type converter. We use this to convert the *Review Text* into text data type. This is necessary for the next operator in the process.
6. **Process Documents**: This is a crucial step in such text classification applications. This is a nested operator, i.e there is a process within this operator as shown below:

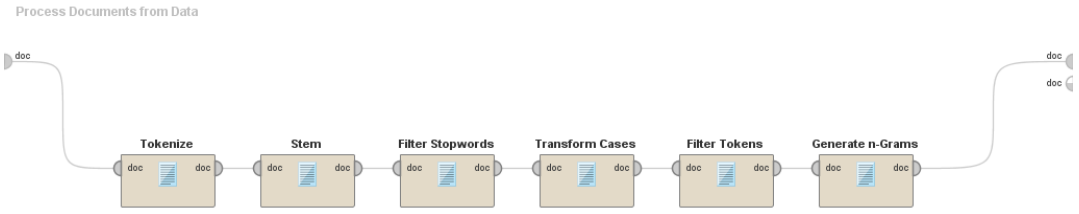

Figure 3 Process flow to generate n-Grams

It breaks down the text into tokenised n-grams, here n was set to 2. These n-grams become part of the bag of words that the model later uses to understand the relationship between the words in the review and the label feature. Tokenizing the document, stemming and filtering out stopwords (like a, an, the etc.) are standard practices. The stemmed tokens are all transformed to lower case and we filter out tokens which have lesser than 5 characters of more than 25 characters. Finally the 2-Grams are generated creating a large bag of phrases. This is the primary data that the model learns from.

7. **Store**: A store operator is used to store the word list generated from the Process documents step. This will have to be used while applying the learnt model on new data. This is an important point to remember.
8. **Split Validation**: The next task is to train and test the model. The Split Validation operator is used for this. This is again a nested operator. It splits the data into training and test data and uses the trained model on the test data. The internal process of this operator is shown below:

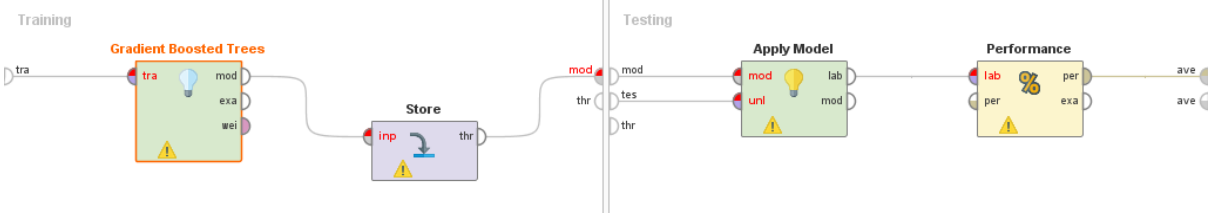

Figure 4 Split validation sub-process

The Gradient Boosted Trees is used here, the trained model is stored for future use using again the Store operator as shown above. The Apply Model operator applies the trained model on the test portion of the data and the Performance Binomial Classification operator is used to measure the performance of the model on the test data. The Performance operator generates a 2x2 table with details of the classification predicted by the model and the actual classification.

Training and evaluation

The next stage after setting up the process in Rapidminer is to try out various models and see what gives good results. This is the part where you actually train your machine learning model.

There are several algorithms that can be tried. It is usually good practice to start with the simple models like k-NN. The k-NN technique is the simplest possible model and can be used in almost all machine learning problems. This helps you set a good benchmark against which you can then try more advanced techniques. With Rapidminer it is as easy as replacing the Gradient Boosted Trees model shown in the figure above with any other predictive model of your choice by right clicking the operator and choosing the replacement. Of course not all models will work for all situations. Some models accept only numerical data. The best way to go about this stage is trial and error, and with practice things will be clear.

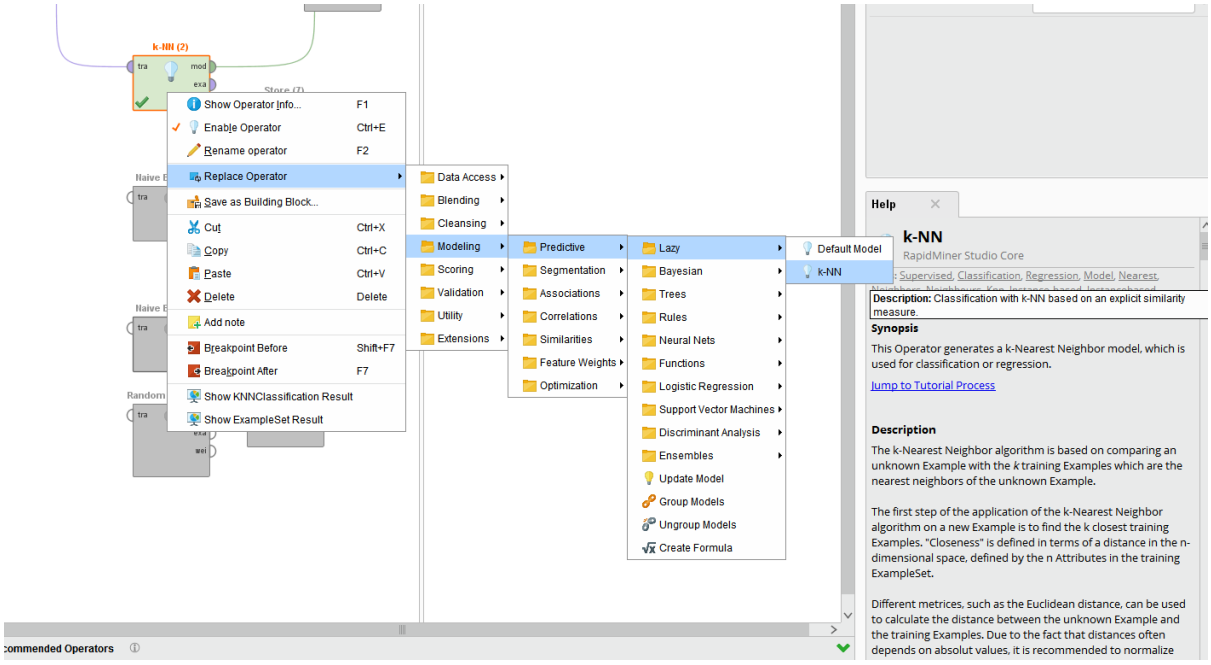

Figure 5 Replacing an operator for trials

The results obtained for the k-NN model is shown below.

accuracy: 62.93%

	true Not trending	true Trending	class precision
pred. Not trending	433	192	69.28%
pred. Trending	225	275	55.00%
class recall	65.81%	58.89%	

Figure 6 k-NN results

Continuing the process the following models were tried and corresponding results of those models are given below.

accuracy: 52.80%

	true Not trending	true Trending	class precision
pred. Not trending	273	146	65.16%
pred. Trending	385	321	45.47%
class recall	41.49%	68.74%	

Figure 7 Naive Bayes results

accuracy: 59.56%

	true Not trending	true Trending	class precision
pred. Not trending	642	439	59.39%
pred. Trending	16	28	63.64%
class recall	97.57%	6.00%	

Figure 8 Decision Tree results

accuracy: 58.49%

	true Not trending	true Trending	class precision
pred. Not trending	658	467	58.49%
pred. Trending	0	0	0.00%
class recall	100.00%	0.00%	

Figure 9 Random Tree results

accuracy: 70.67%

	true Not trending	true Trending	class precision
pred. Not trending	464	136	77.33%
pred. Trending	194	331	63.05%
class recall	70.52%	70.88%	

Figure 10 Gradient Boosted Trees' results

The Gradient Boosted Trees (GBT) gave the best results of all models tried. The trial and error was done with a subset of the full dataset. Now we can reuse the learnt GBT model and apply it for the full dataset and check how the results are. The process to retrieve the stored GBT model and associated word list is shown below, it is important to use the same word list as was generated during the training of the model. This step helps to check that the learnt model has not over fit the training data; if it does over fit, then it will not be very useful for real time application.

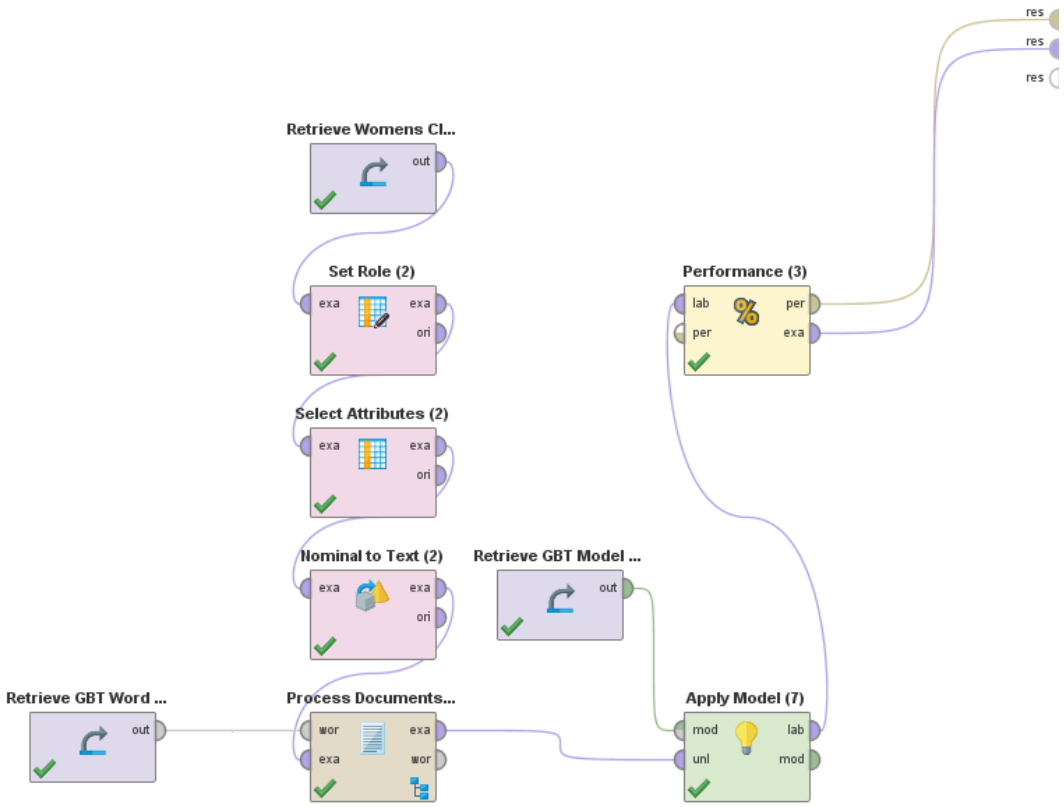

Figure 11 Applying the learnt GBT model to the full dataset

The GBT results for the larger dataset is given below:

accuracy: 72.40%

	true Trending	true Not trending	class precision
pred. Trending	2746	1179	69.96%
pred. Not trending	1581	4494	73.98%
class recall	63.46%	79.22%	

Figure 12 GBT results for wider dataset

Being able to predict the chances of success of products like this can be very useful in planning inventory and distribution. Perhaps success rate across demographics and geographic locations can be predicted, that can be very useful in planning for the right distribution of products. As with any model, it is important to test it further by running it in parallel with live data to further optimize the accuracy of the model. Such applications clearly show the utility of such techniques and their prevalence is only going to increase.

Once the trained model and word list are stored it can be reused for application to new data. This would involve obtaining the new products' review and rating data in the same format as in training data and replace the first operator, in the process shown above for applying the learnt GBT model, with the new data. Run the process and see what the predictions are. A category manager handling a wide array of products can run this process at a set frequency, the system will help point him in the right direction and in a matter of only few minutes. Of course there are ways to output the predicted

data into an excel file that can be circulated to the team to help them take decisions on the supply chain aspects of their products.

Of course there are still ways to further improve on the system. Perhaps instead of mathematically defining successful products, if data can be mined that provides the actual success or failure of the product the accuracy can be improved. In this tutorial we have created a technique based on available data, one can even try to source rich data knowing the process that can be used, this will also require some work on enabling such data creation and collection, but could be the most accurate.

In summary, we have implemented a quick model based on NLP that predicts a product's chances of success based on text reviews of the product. Such a system can help a category manager handle a large number of products and SKUs. The time consumed to apply the model on new data is just a matter of minutes or even seconds and this can be taken advantage of at a high frequency.