SENTIMENT ANALYSIS

IDS 572 – Assignment 4

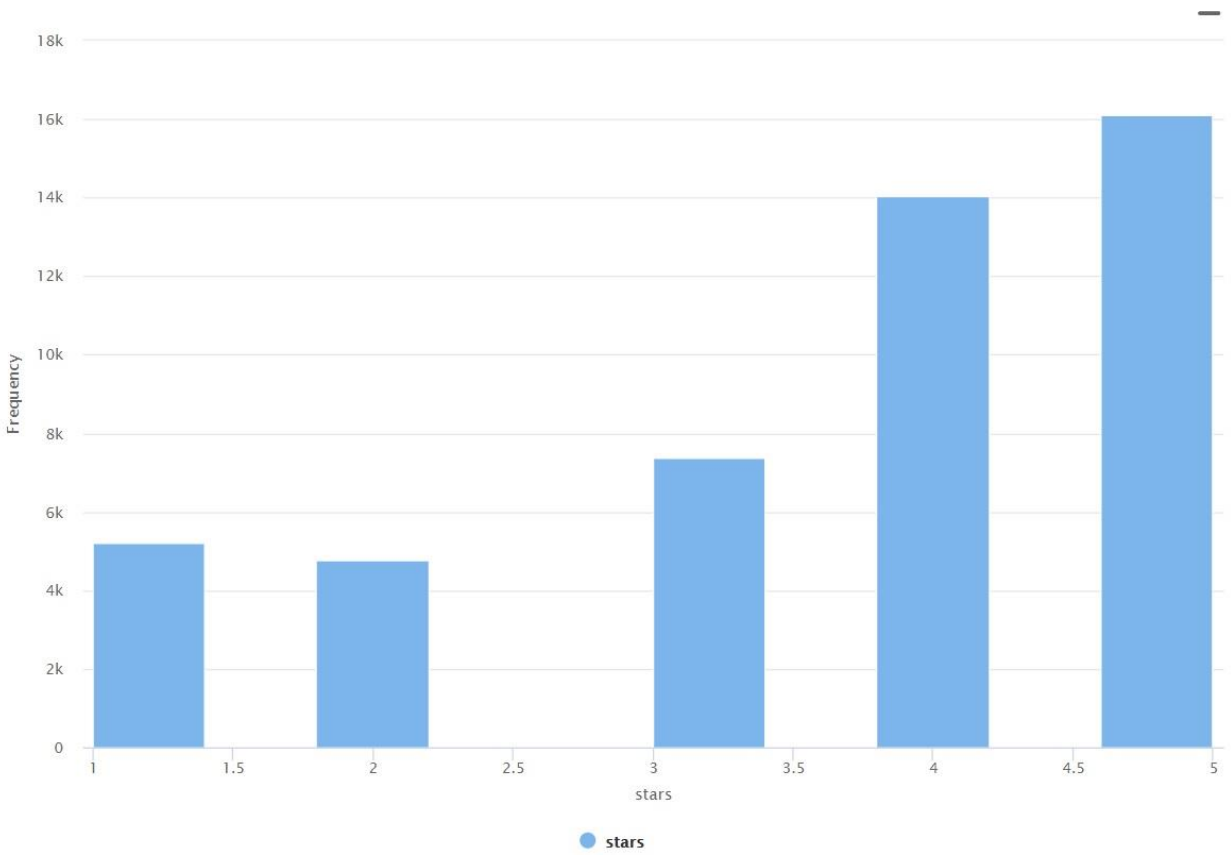
Conducted Text Mining of user-review data and sentiment analyses based on a collection of reviews and accompanying star ratings on Yelp. Used a sample of the original dataset to examine the effectiveness of different sentiment 'dictionaries'. Developed and evaluated classification models to help predict sentiment polarity.

BUSHRA PATNI
660532615

TALISH BARMARE
662161711

NISHANTH S
655497371

(a) Explore the data. How are the star ratings distributed? How will you use the star ratings to obtain a label indicating 'positive' or 'negative' – explain using the data, graphs, etc.? Do star ratings have any relation to 'funny', 'cool', 'useful'?



The above graph depicts the "star ratings" and observe the following:

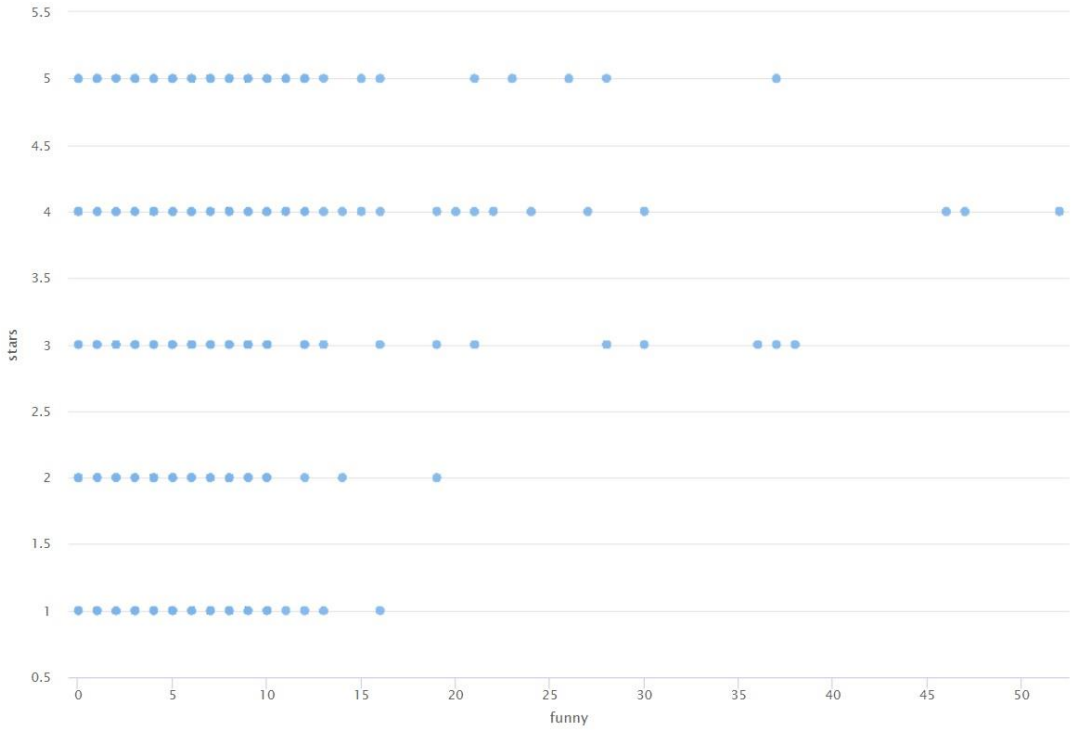
- The range of star ratings is from 1.00 to 5.00.
- The maximum number of star ratings are between 4.60 and 5.00 i.e. 16,091.
- The minimum number of ratings are between 1.00 and 1.40 i.e. 5,224.

The actual count for each of the rating levels is given in table below:
Minimum -1, Maximum -5, Average - 3.635, Standard Deviation: 1.328

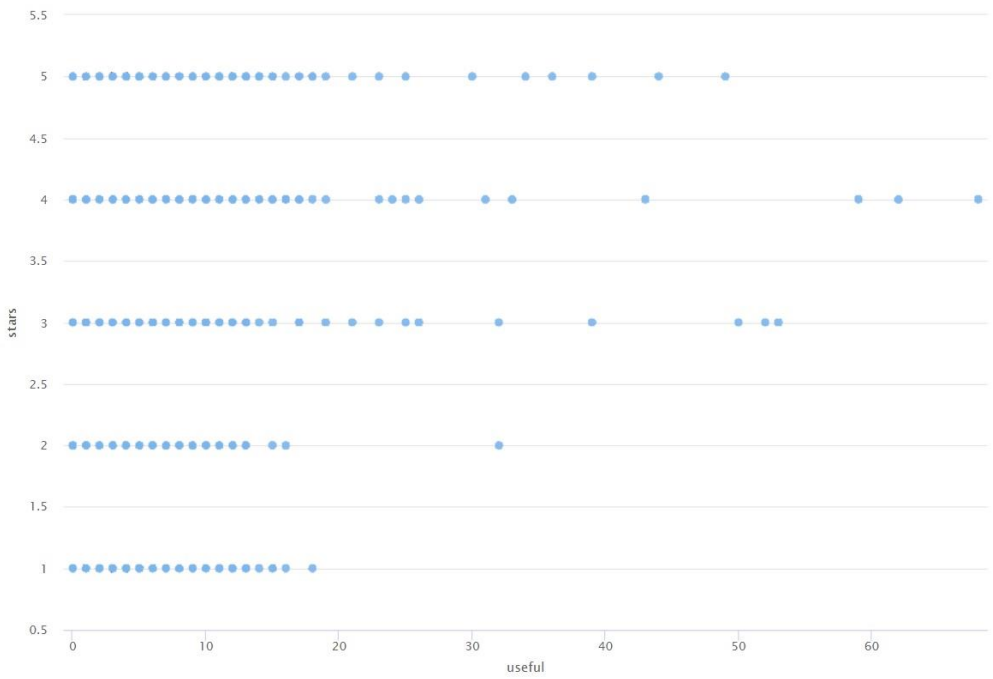
Rating bands	**1**	**2**	**3**	**4**	**5**
Distribution	5,224	4,756	7,381	14,042	16,091

On average, the rating is 3.635. Majority of the ratings are covered by ratings 4 and 5 i.e. approximately 63.45%. Anything over 3.635 shall be taken as positive whereas anything below 63.45% shall be taken as negative reviews.

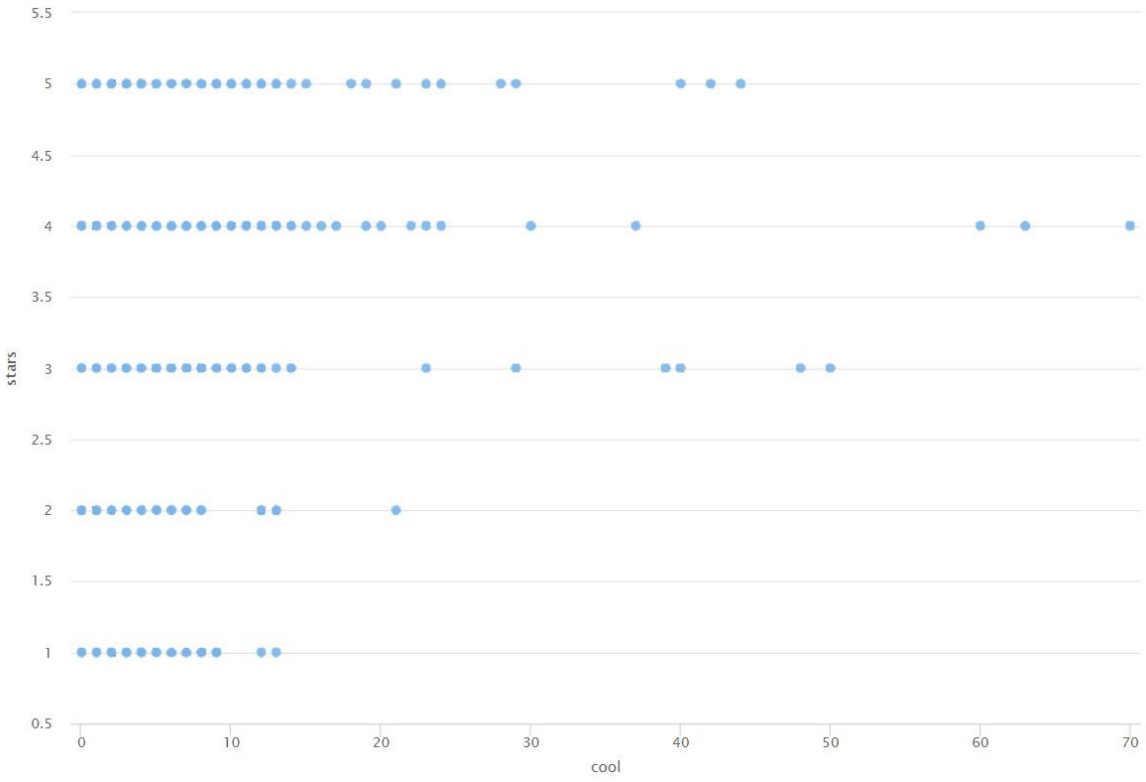
Below are the visualizations of ‘funny’, ‘cool’ and ‘useful’:



Funny: Observing the above plot, we see that the word ‘funny’ is mentioned in all star ratings. The maximum number of times it has been mentioned is in star rating ‘4’. It is highest at 52 4-star reviews.

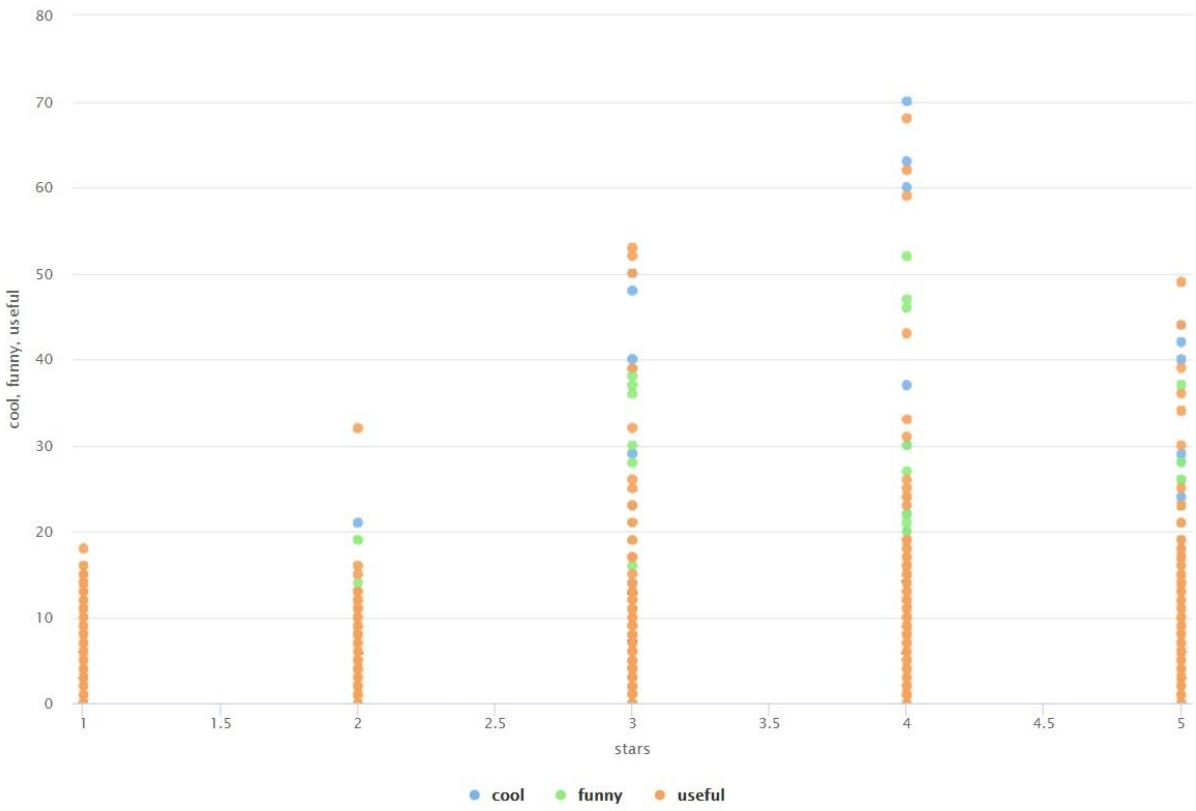


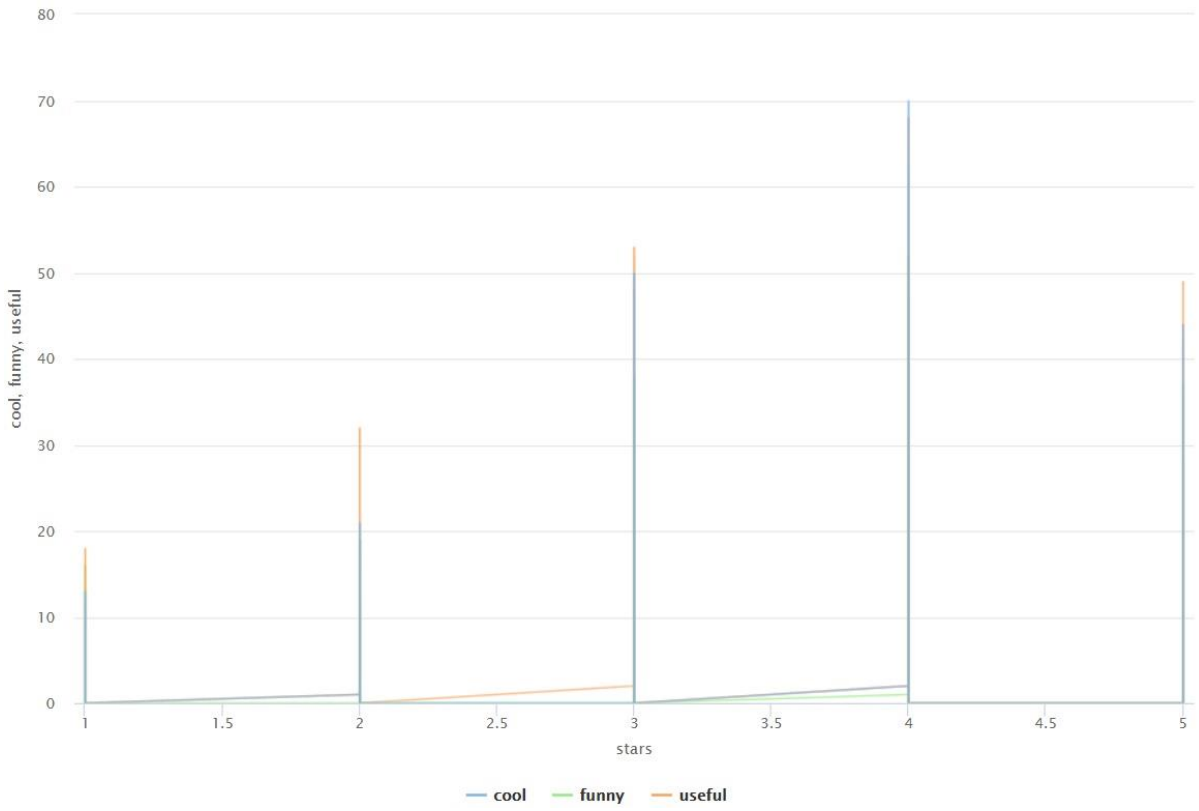
Useful: Useful appears in a lot of 4-star ratings, with 68 being the highest.



Cool: In a lot of 4-star ratings, cool appears with 70 being the highest.

The below scatter and series plots by overlaying all 3 categories of review “words” in one chart with respect to star ratings.





Useful can be categorized as neutral as it has most occurrences at 3.0. On the other hand, Cool and Funny seem to share distributions. Taking intuition into account, we anyway tend to look at reviews with words as funny and cool which are positive in nature. Thus, this falls with our line of thought.

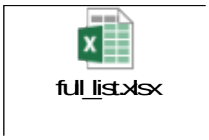
(b) What are some words indicative of positive and negative sentiment? Do these 'positive' and 'negative' words make sense in the context of user reviews?

The table below gives us a list of Top 10 words seen in reviews. The words listed below are the ones user would associate with when they look at reviews.
In Document: # of documents the token occurs in; **Total**: # of documents; **Value**: average star rating for each word.

Row No.	word	in docu... ↓	total	att_1
196	food	22876	34433	0.816
227	good	18865	28791	0.829
411	place	18448	26671	0.779
496	service	14436	16732	0.567
231	great	14340	19742	0.817
570	time	9670	12618	0.611
216	get	9520	12399	0.611
451	restaurant	8267	11061	0.639
381	ordered	7375	9856	0.570
349	nice	7055	8669	0.609
75	chicken	6893	10750	0.809
121	delicious	6696	7759	0.678
380	order	6603	8771	0.566
587	try	6454	7573	0.572
332	menu	6400	8161	0.606

ExampleSet (647 examples, 0 special attributes, 4 regular attributes)

Based on the average star rating for each word, we have obtained a list of positive and negative words. From below, we have listed values of a few positive and negative words. We have attached a csv file below which contains the full list.



Positive words:

Word	in documents	total	att_1
yum	625	764	0.84162
good	18865	28791	0.828913
great	14340	19742	0.817379
love	5624	6902	0.745894
incredible	548	584	0.737059
gem	571	579	0.72769
excellent	3132	3501	0.71209
outstanding	609	658	0.710329
amazing	4438	5084	0.705599
wonderful	1384	1541	0.704697

Negative words:

Word	in documents	total	att_1
worst	957	1036	0.299113
terrible	855	939	0.333371
rude	697	780	0.337807
horrible	823	909	0.340262
disappointing	637	665	0.349795
poor	602	678	0.357167
mediocre	705	724	0.384137
bland	1224	1362	0.387902
sorry	511	530	0.40018
waited	949	1112	0.408295

Again, through intuition, as customers, we would naturally associate similar words with good and bad experiences. When we compare the words, we generated with how reviews with low rating and reviews with high rating are structured, it makes complete sense.

(c) We will consider three dictionaries – the Harvard IV dictionary of positive and negative terms, the extended sentiment lexicon developed by Prof Bing Liu of UIC-CS, and the AFINN dictionary which includes words commonly used in user generated content in the web. Details on these are given below. As discussed in class, the first two provide lists of positive and negative words, while the third gives a list of words with each word being associated with a positivity score from -5 to +5. How many matching terms are there for each of the dictionaries?

(i) Consider using the dictionary based positive and negative terms to predict sentiment (positive or negative based on star rating) of a movie. One approach for this is: using each dictionary, obtain an aggregated positiveScore and a negativeScore for each review: for the AFINN dictionary, an aggregate positivity score can be obtained for each review. Are you able to predict review sentiment based on these aggregated scores, or how do they perform? Does any dictionary perform better?

There are 3 dictionaries we have taken: The Harvard Dictionary, the AFINN and Bing Liu Dictionary. Of these, The Harvard Dictionary has about 1636 positive and 2006 negative words, the AFINN has 878 positive words and 1598 negative words and the Bing Liu Dictionary has 2006 positive and 4783 negative words.

The output observations for all the three different dictionaries are tabulated below Dictionary	**No. of words Matched**
Harvard IV positive; Harvard IV negative	22
Sentiment Lexicon positive; Sentiment Lexicon negative	104
Afinn Positive; Afinn Negative	94

We are using percental pruning, below 1% and above 98% are pruned.

The prediction accuracy and performance in number of words for each dictionary is given below:

Dictionary	Accuracy
Bing Liu	47.63%
Harvard	45.91%
AFINN	43.87%

Prof Bing Liu's Lexicon Sentiment dictionary seems to perform the best.

(ii) Compare this approach with use of the SentiWordNet. Describe how you use SentiWordNet.

First, it is important to understand what is SentiWordNet exactly.
SentiWordNet is a lexical resource used for opinion mining that assigns to each synset of WordNet three sentiment scores: positivity, negativity, and objectivity. It has a web-based graphical user interface, freely available for research purposes. The development of the resource is based on the quantitative analysis of the glosses associated to synsets, and on the use of the resulting vectoral term representations for semi-supervised synset classification. By combining the results produced by a committee of eight ternary classifiers, positivity, negativity, and objectivity are derived.

On applying the SentiWordNet lexical resource to the problem of automatic sentiment classification of Yelp reviews, our approach comprises counting positive and negative term scores to determine sentiment orientation, and an improvement is presented by building a data set of relevant features using SentiWordNet as source and applied to a machine learning classifier. **We find that results obtained with SentiWordNet are in line with similar approaches using manual lexicons like Harvard IV, AFFIN & Sentiment Lexicon.**

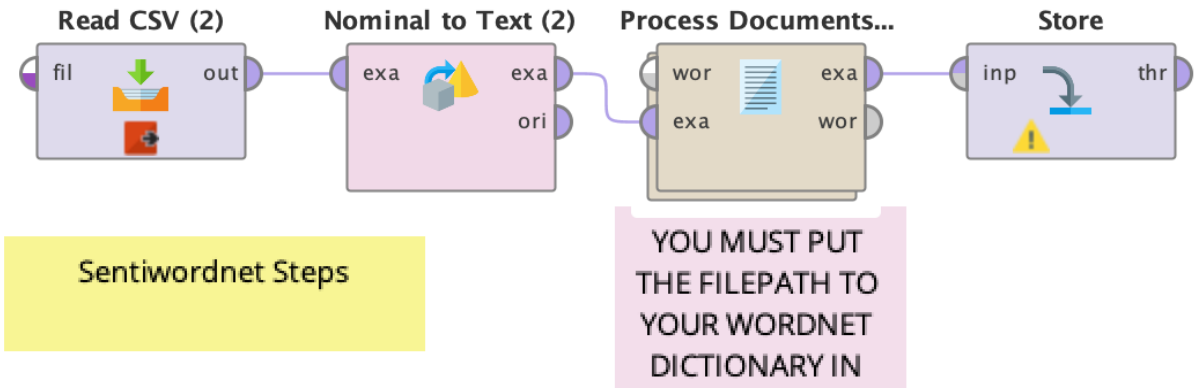
How to use SentiWordNet:

There are several methods provided by the Rapidminer text mining capabilities for Sentiment analysis. Using the wordnet dictionary and relevant operators from Rapidminer and Wordnet Dictionary are one of the popular methods when dealing with English text.

There are certain essential prerequisites needed which are as follows:

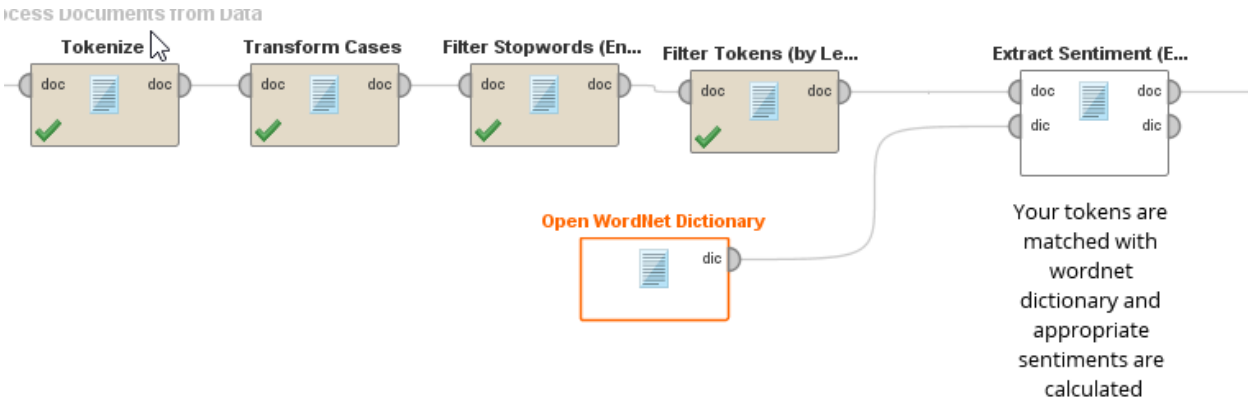
- Wordnet Extension
- Text Processing Extension
- The wordnet dictionary

In the screen shot below we are reading the original data set, then changing data type of the column we want to use for "text processing" and then passing the dataset (Exampleset) to "Process Documents from Data".



We add standard text processing steps like tokenize, transform cases, filter stops words, filter tokens etc.

Then the two operators; to get the sentiment score are "Open WordNet dictionary" and "Extract Sentiment (English) both coming from the Wordnet extension.



This process adds a new column **"sentiment"** that provides a numeric value for sentiment, Negative sentiment are scored less than zero and positive sentiments are code greater than zero.

busines...	CAuRuA...
cool	1
funny	1
review_id	TM_Faa...
stars	review_id
useful	1
sentiment	0.234

One can use the sentiment score and "Generate Attributes" operator to flag documents as Positive, Neutral, Negative etc. based on the actual score value itself.

(d) Develop models to predict review sentiment. For this, split the data randomly into training and test sets. To make run times manageable, you may take a smaller sample of reviews (minimum should be 10,000).

(i) Develop models using only the sentiment dictionary terms (you can try individual dictionaries or combine all dictionary terms).

Solution:

i. For all 3 dictionaries, the output document term matrix is used as inputs in this process.

ii. The next step is we are generating new attribute (predictor variable: sentiment). We are expressing this function that if (stars>3," true", "false") i.e. if stars are above 3.0 we are predicting the sentiment as true else we are predicting it as false.

iii. We are using select attribute and selected all the words from the input document term matrix and for our output label, we are selecting sentiment.

iv. We are generating TF_IDF which Compute TF-IDF values across all words and reviews in the Document Term Matrix. We use TF-IDF over term frequency since its lesser time consuming to run the RapidMiner process than computing term frequency.

v. We have taken a smaller sample of reviews around 25% i.e., 11,744 samples to make run times manageable.

vi. We are splitting the data into two sets of training data and testing data with the train to test ratio as 0.8:0.2.

vii. Then, to predict our target variable which is sentiment, we have used different models on training data and we have applied the model results to our test data.

Models Used:

We have used 3 models to predict our target variable which are:

1. Logistic Lasso Regression: We used Lasso over ridge regression because we observed that it has better generalization performance and it also has lesser tendency to over-fit when compared to the ridge logistic regression.
2. Naïve Bayes
3. KNN model

The best model overall is Logistic Regression using Sentiment Lexicon with Lambda as 0.01 and alpha = 0.001. It gives us an accuracy of 63.13% on Test Data.

Sentiment Lexicon: Final values:

For Sentiment Lexicon, the best model is Logistic Regression with Lambda as 0.01 and alpha = 0.001. It gives us an accuracy of 63.13%.

Model	**Parameters**	**Training Accuracy**	**Testing Accuracy**
Logistic Regression	Lambda= 0.5, alpha =0.1	45.51%	43.98%
	Lambda= 0.01, alpha =0.001	**64.53%**	**63.13%**
	Lambda= 0.01, alpha =0.01	61.51%	59.05%
	Lambda= 0.01, alpha =1	58.07%	55.43%
Naive Bayes	With Laplace Correction	84.08%	59.30%
	Without Laplace Correction	63.01	61.95
K-NN	k=3	91.13%	58.71%
	k=5	72.52%	59.94%
	k=7	70.55%	59.73%

AFINN:

Model	**Parameters**	**Training Accuracy**	**Testing Accuracy**
Logistic Regression	Lambda= 0.5, alpha =0.1	45.51%	43.98%
	Lambda= 0.0001, alpha =0.001	65.98%	62.20%
	Lambda= 0.01, alpha =0.01	58.09%	56.02%
	Lambda= 0.01, alpha =1	57.73%	55.39%
Naive Bayes	With Laplace Correction	61.31	58.54%
	Without Laplace Correction	61.31	60.93
K-NN	k=3	71.59%	55.73%
	k=5	71.07%	56.58%
	k=7	73.05%	59.90%

Harvard

Model	Parameters	Training Accuracy	Testing Accuracy
Logistic Regression	Lambda= 0.5, alpha =0.1	45.51%	43.98%
	Lambda= 0.01, alpha =0.001	56.25%	57.13%
	Lambda= 0.01, alpha =0.01	50.08%	49.34%
	Lambda= 0.01, alpha =1	49.26%	48.45%
Naive Bayes	With Laplace Correction	65.34%	58.91%
	Without Laplace Correction	62.10%	49.49%
K-NN	k=3	51.82%	48.79%
	k=5	53.81%	48.74%
	k=7	52.91%	48.91%

Do you use term frequency, tfidf, or other measures? What is the size of the document-term matrix?

To reduce the effect of words that occur multiple times in one review, we have used TF-IDF for running our models. The size of the matrix is (11744,107).

(ii) Develop models using a broader list of terms – how do you obtain these terms? Will you use stemming here? Report on performance of the models. Compare performance with that in part (c) above. For models in (i) and(ii): Do you use term frequency, tfidf, or other measures, and why? Do you prune terms, and how (also, why?). What is the size of the document-term matrix?

The broader list of terms was obtained by lemmatizing and increasing the range of the stars. We are not using stemming in this process as it will reduce our results if we are using a dictionary. In the process, we had a lot of terms. So, we used absolute pruning to reduce model complexity and set the pruning values below absolute value 40 and above absolute value 90. The pruned models look more stable and variance is less in contrast to unpruned models. The document matrix size was [20000, 147].

Model	**Parameters**	**Vector creation**	**Training Accuracy**	**Testing Accuracy**
Naive Bayes	With Laplace	TF-IDF	69.58%	28.91%
	With Laplace	TF	69.56%	29.34%

Model	**Parameters**	**Vector creation**	**Training Accuracy**	**Testing Accuracy**
k-n-n	K=3	TF-IDF	82%	76.49%
	k=3	TF	83.45%	75.21%
	k=7	**TF-IDF**	**86.76%**	**78.94%**
	k=7	TF	84.55%	78.67%

By comparing the results from (c) our best model is knn which has performed much better and has given much better training and testing accuracies. We preferred TF-IDF over TF because we are joining and comparing words from three different dictionaries and we wanted to normalize the term frequencies from all the three dictionaries. Apart from this, the performance of TF-IDF was more than TF.