Text Mining: Sentiment Analysis of US Airlines Tweets

Stuti Sharma
Seattle Pacific University

Abstract

Social media is a great platform for the customers to express their views, opinions, and sentiments. Twitter is one of the widely used social websites to shape the opinion of the people and influence the brand's perception. The goal of the project is to build a model for the two major U.S Airlines: Delta and Southwest, to perform sentiment analysis on the customer's tweets using the Rapid Miner tool. By doing so, the airlines can gain an insight of the wider public opinion behind certain positive and negative tweets. This will enable airlines to quickly understand customer attitudes and react accordingly to leverage competitive advantage. The Twitter U.S Airlines dataset has been taken from Kaggle website, which consists 2932 tweets with positive and negative polarities. In the paper, Decision Tree has a maximum accuracy of 76.43 % among the various other algorithms like Random Tree, Naïve Bayes, and K-NN. Towards the end, I have also performed clustering and association analysis on word vectors to identify similar word vectors and to find common themes that occur frequently across the word vectors respectively.

Keywords: Twitter, U.S. airlines, sentiment analysis, cluster, association, Naïve Bayes, RapidMiner, Decision Tree, K-NN

Introduction

<u>Objective:</u> The objective of the project is 1) to use machine learning algorithms to predict whether the tweet is “positive” or “negative” by analyzing the feelings of the passengers. 2) Determine the best model to assign the sentiment (“positive” or “negative”) to the test data.

<u>Dataset:</u> The Twitter U.S. Airlines datasets have been taken from the Kaggle.com (https://www.kaggle.com/crowdflower/twitter-airline-sentiment/version/2). Kaggle is the platform for predictive modeling and analytics competitions where data miners upload various datasets to compete and produce the best models. This dataset is the reformatted version of the original source (*Crowdflower's Data for Everyone library*.) retrieved in February 2015 from Twitter. The dataset contains tweets with the sentiment set as “Positive” and “Negative” for two major U.S. Airlines: Delta and Southwest. The dataset consists of 2932 rows with 5 independent attributes like Passenger Name, Airline, Tweet, Retweet Count, Passenger Time Zone, anSentiment (label). There are 38% Positive Tweets and 62% Negative Tweets in the dataset.

<u>Tool:</u> I have chosen Rapid Miner Studio 8.1 for the text mining and sentiment analysis because it has more than 100 learning operators and easy framework for the beginners like effortless installation, attractive and clear user interface. Rapid Miner has enormous flexibility in process designs e.g. dragging and dropping of the operators in the process window.

- Rapid Miner Auto Model suggests various operators to the user based on the processes used by other data miners to build the model.
- The user can visualize the end to end data preparation and modeling steps by seeing the simple statistics, charts, tables, and graphs.
- Rapid Miner can read and load various types of data formats like excel, csv, arff, XML, HTML etc.
- The user can download various extensions from Rapid Miner Marketplace for free.

<u>Procedure:</u>

Step_1: Open RapidMiner → Extensions → Marketplace → Top Downloads → Select (Text Processing, Operator Toolbox, and Web Mining) → Install. Can be seen from the Figure 1,2 and 3).

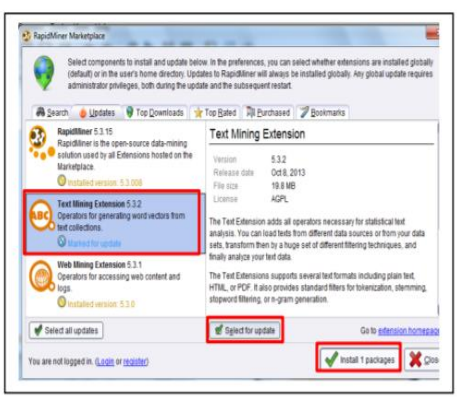


Figure_1

Figure_2

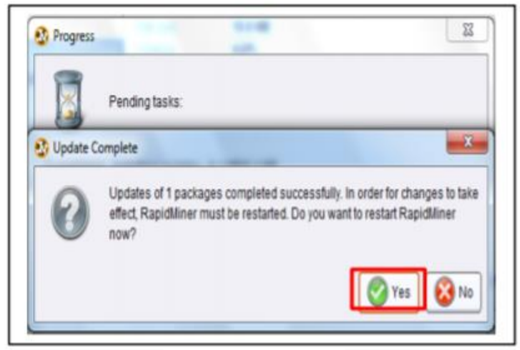


Figure_3

Step_2: Added the excel dataset (U.S. Airline Tweets.xlxs) into the RM Local Repository from the computer, for the easy retrieval (Fig 4, 5, 6). Then, dragged the "Retrieve Operator" from the Operator Window to the Process Window and opened the dataset file in the Parameter Window (repository entry) as shown in the snapshot (Fig 7).

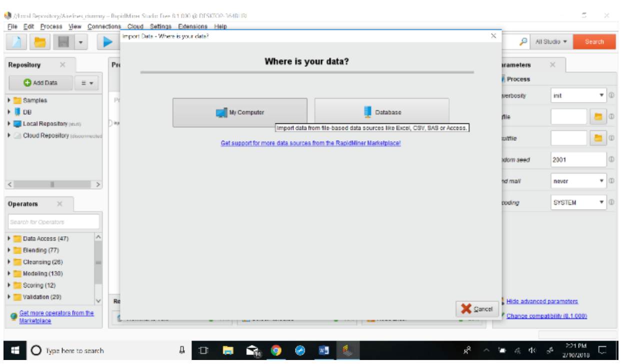


Figure_4

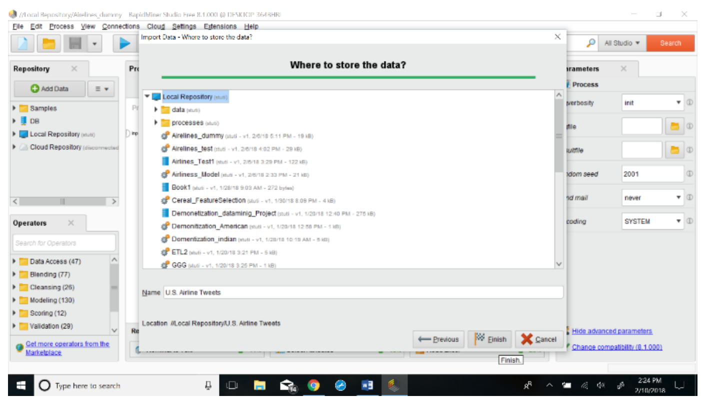


Figure_5

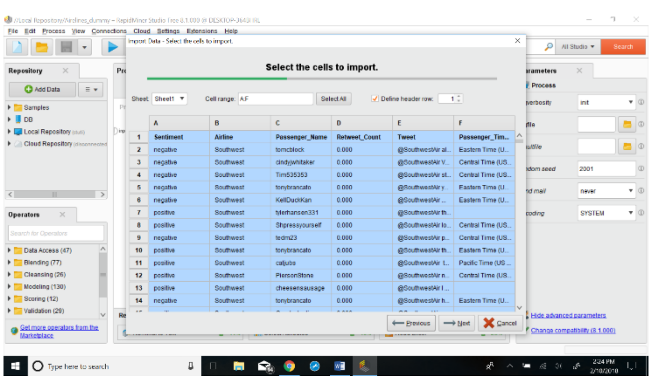


Figure_6

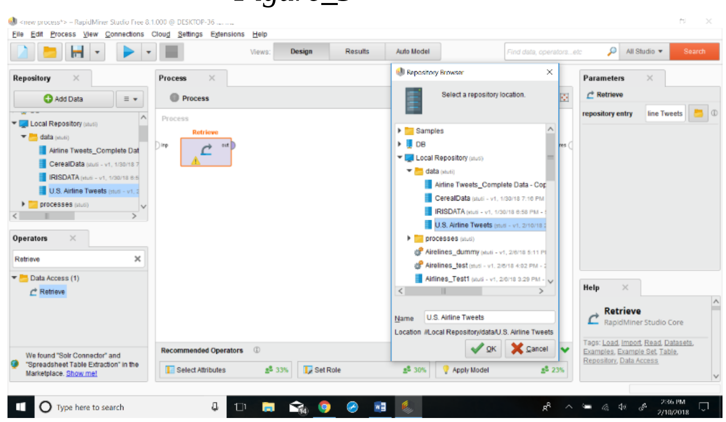


Figure_7

Step 3: Exploring Trends

From figure 8 we can interpret that Southwest airline got slightly more tweets (1500 tweets) than Delta (1432). Both the airlines have a higher number of negative tweets than positive tweets. If the sentiment is classified accurately, this will help the airlines to locate the cause behind passengers' negative tweets and improve their services overall. Figure 9, we can conclude that if the sentiment is predicted accurately, passengers are more likely to retweet a positive tweet than a negative tweet. Figure 10, we observed that the passengers who do not mention the location are more likely to tweet a negative tweet and most of the tweets originate from the Eastern Time Zone (U.S. & Canada).

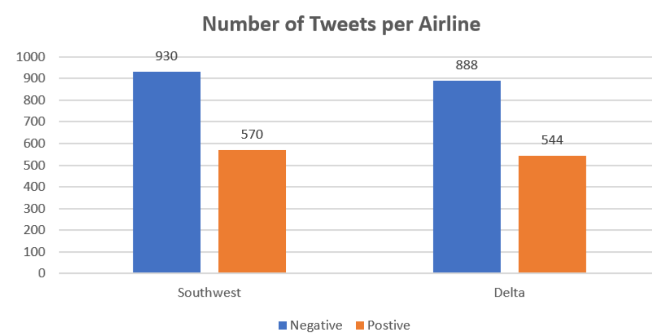


Figure_8

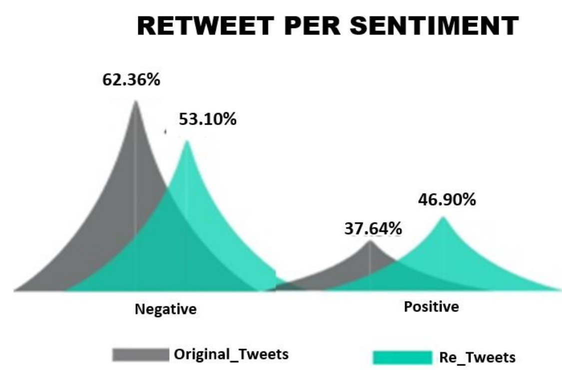


Figure_9

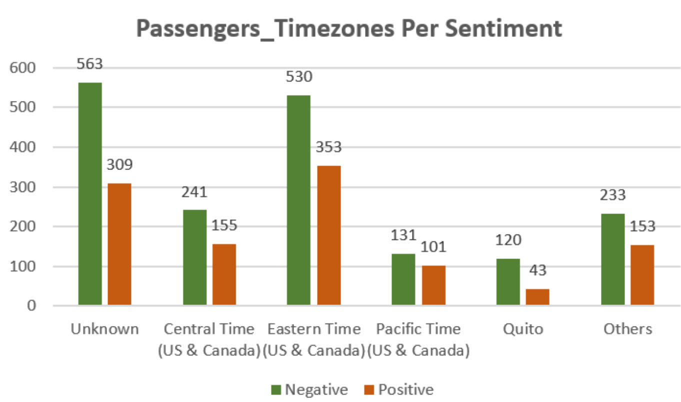


Figure_10

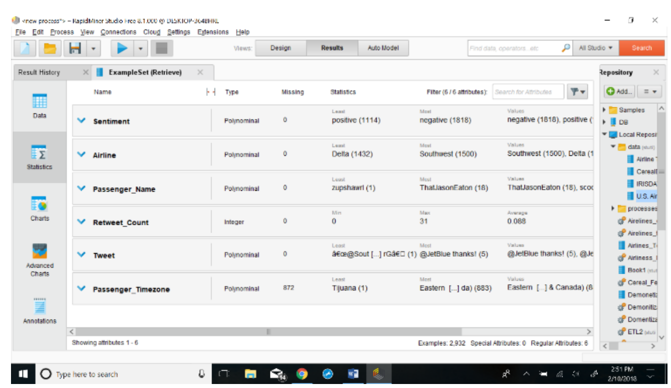
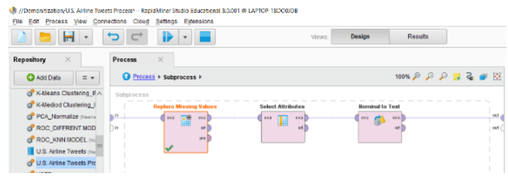


Figure_11

Step_4: Data Preparation



1. Select & drag "Subprocess" form the Operator Window and connect it to "Retrieve" in Process Window. Subprocess Operator introduces processes within a process. Following operators are used inside the "Subprocess":
   - Replace Missing Value: This operator is used to replace all the missing values of Passenger_Timezone attribute by replenishment value "Unknown" (Figure 11 above)

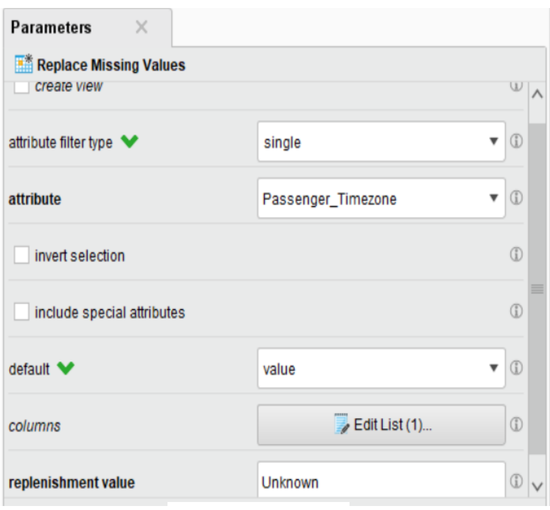


Figure_12

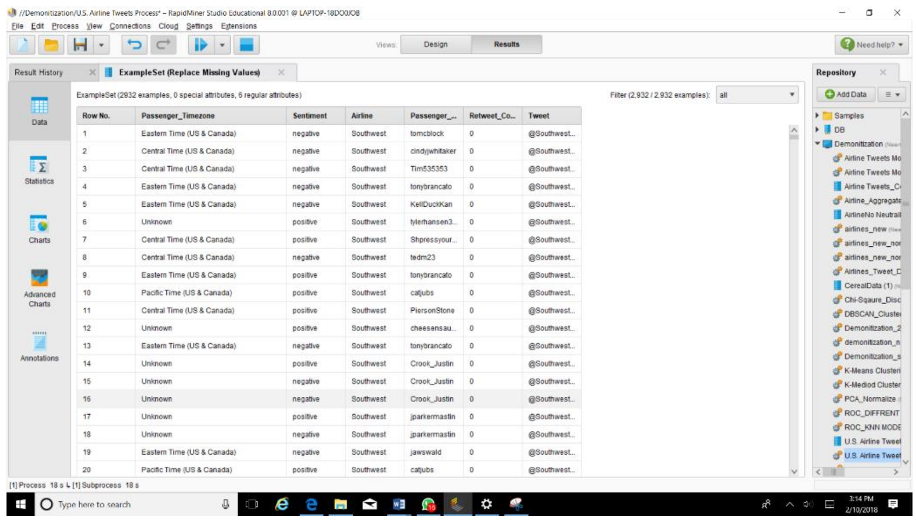


Figure_13

   - Select Attribute: This operator is used to select a subset of attributes (Sentiment and Tweets) which are relevant to the project objective and remove the other attributes.
   - Nominal to Text: Since, Process Document Operator works only on text data type, which is going to be used in further data cleaning we must change the nominal attributes like Tweet to string data type.

2. Select & drag 2nd "Subprocess" form the Operator Window and connect it to 1st Subprocess Operator in Process Window. Following operators are used in this process:
   - Replace: This operator replaces part of the values of text matching a specified regular expression (@, HTTP?!, #) by a specified replacement (attag, linktag,

questiontag, exclamationtag, hashtag). We have used this operator to simply the text data for easy extraction of polar words. (Figure 14)

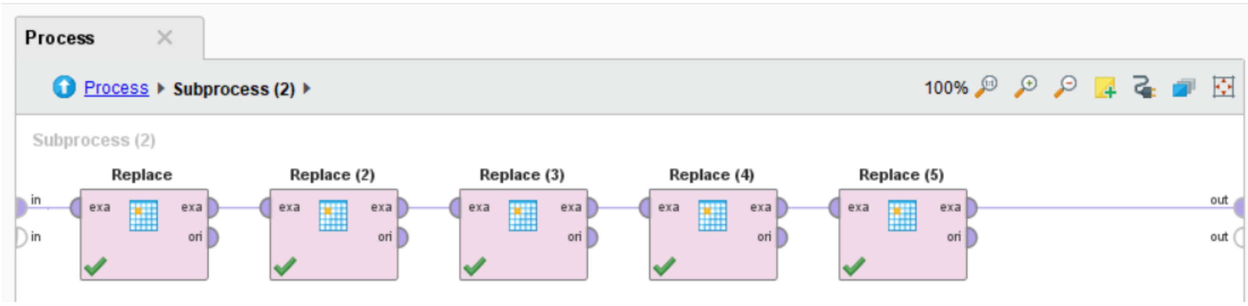
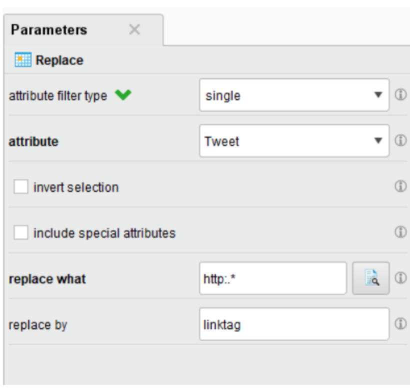
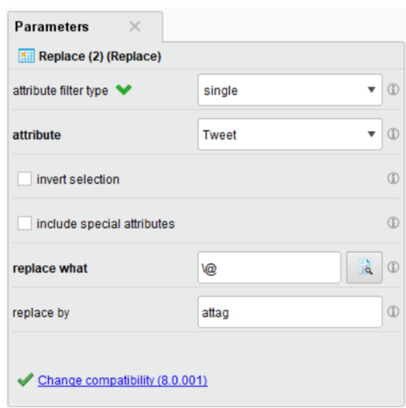
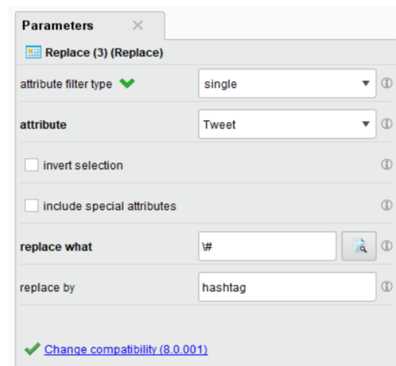
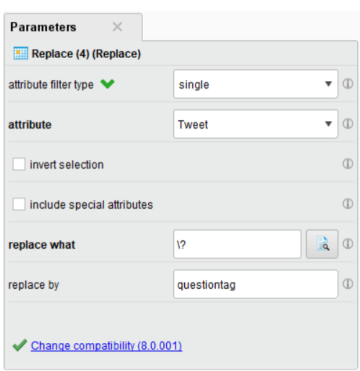


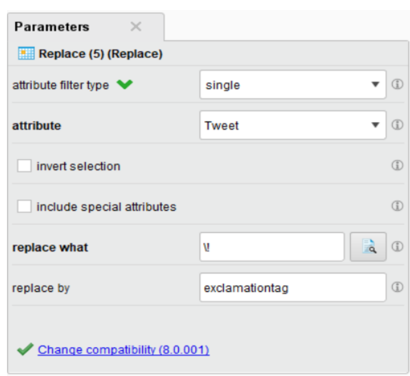
Figure_14

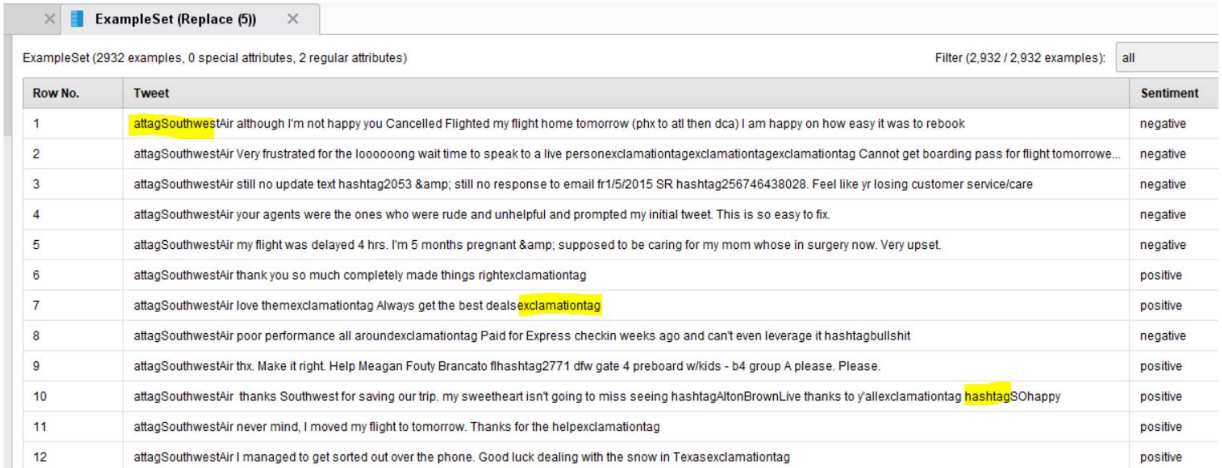












ExampleSet (Replace (5))

ExampleSet (2932 examples, 0 special attributes, 2 regular attributes) Filter (2,932 / 2,932 examples): all

Row No.	Tweet	Sentiment
1	attagSouthwestAir although I'm not happy you Cancelled Flighted my flight home tomorrow (phx to atl then dca) I am happy on how easy it was to rebook	negative
2	attagSouthwestAir Very frustrated for the loooooong wait time to speak to a live personexclamationtagexclamationtagexclamationtag Cannot get boarding pass for flight tomorrowe...	negative
3	attagSouthwestAir still no update text hashtag2053 & still no response to email fr1/5/2015 SR hashtag256746438028. Feel like yr losing customer service/care	negative
4	attagSouthwestAir your agents were the ones who were rude and unhelpful and prompted my initial tweet. This is so easy to fix.	negative
5	attagSouthwestAir my flight was delayed 4 hrs. I'm 5 months pregnant & supposed to be caring for my mom whose in surgery now. Very upset.	negative
6	attagSouthwestAir thank you so much completely made things rightexclamationtag	positive
7	attagSouthwestAir love themexclamationtag Always get the best dealsexclamationtag	positive
8	attagSouthwestAir poor performance all aroundexclamationtag Paid for Express checkin weeks ago and can't even leverage it hashtagbullshit	negative
9	attagSouthwestAir thx. Make it right. Help Meagan Fouty Brancato flhashtag2771 dfw gate 4 preboard w/kids - b4 group A please. Please.	positive
10	attagSouthwestAir thanks Southwest for saving our trip. my sweetheart isn't going to miss seeing hashtagAltonBrownLive thanks to y'allexclamationtag hashtagSOhappy	positive
11	attagSouthwestAir never mind, I moved my flight to tomorrow. Thanks for the helpexclamationtag	positive
12	attagSouthwestAir I managed to get sorted out over the phone. Good luck dealing with the snow in Texasexclamationtag	positive

Figure_15: Result of the Replace Operators in Subprocess_2

3. Select & Drag "Process Document from Data" Operator and connect to 2nd Subprocess Operator in the Process Window. This operator generates word vectors from the string attribute i.e. Tweet to remove any unwanted terms. It also introduces a process within a process. Following processes are carried out in this operator (figure 16).
   - Transform Cases: This operator transforms all characters to one case for simplification purpose. We have chosen a lower case.
   - Tokenize: This operator splits the text of the tweets into a sequence of tokens. Select the mode as "non-letter" for splitting the text into word token whenever it is non-letter like a full stop, space, numerical value etc.
   - Filter Token (by Length): It removes the word shorter than or longer than the configured number of characters. Our range is 3-25 characters in a token.
   - Filter Stopwords (English): This operator removes English stop words from the document like then, that, the, etc.
   - Filter Token (by Content): This operator filters tokens based on their content. We want to exclude all the tokens present in the document that contains the word "tag" e.g. hashtag, linktag, attag etc. by selecting "invert condition" and providing a regular expression as ".*tag.*".
   - Stem (Porter): This operator is used to remove suffix of the words. By doing this we can reduce the number of words and can have accurately matching stems e.g. the words happy, happier and happiest all can be stemmed to the word "happy".

Set Role operator is used after the Process Document operator to select the label i.e. Sentiment which will act as the target attribute for the learning operators.

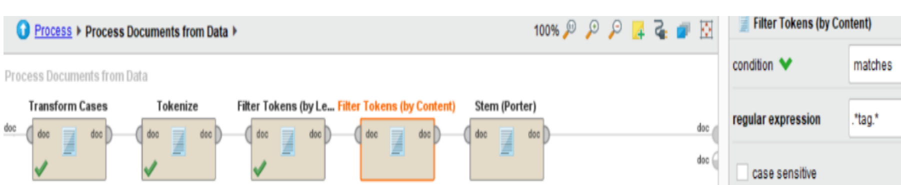


Figure_16

Once we apply all the operators in process document from data, for all the tweets in the dataset, a document by term matrix will be generated. Below is the example set from “process document from data” operator (figure 17)

Row No.	Sentiment	accept	access	accid	accommod	accomplish	accord	account
164	positive	0	0	0	0	0	0	0
165	negative	0	0.335	0	0	0	0	0
166	positive	0	0	0	0	0	0	0
167	negative	0	0	0	0	0	0	0
168	negative	0	0	0	0	0	0	0
169	negative	0	0	0	0	0	0	0
170	negative	0	0	0	0	0	0	0
171	negative	0	0	0	0	0	0	0
172	positive	0	0	0	0	0	0	0
173	positive	0	0	0	0	0	0	0
174	negative	0	0	0	0	0	0	0
175	positive	0	0	0	0	0	0	0
176	positive	0	0	0	0	0	0	0
177	negative	0	0	0	0	0	0	0
178	negative	0	0	0	0	0	0	0
179	positive	0	0	0	0	0	0	0
180	negative	0	0	0	0	0	0	0.254

Figure_17

Here we can see the documents as rows and the terms as columns. We can also observe that the TF-IDF scores are assigned to each term across all the documents. The TF-IDF shows how important a word is to the document in the whole corpus. Hence it is mainly used as a weighing factor in text mining scenario. TF stands for term frequency, i.e. it gives a frequency of terms in a document and IDF stands for inverse document frequency. It shows how informative a word is across all the document. For example, in the above figure, the TF-IDF score of the word “access” is 0.335 in document 165 and for the word “account”, the TF-IDF score is 0.254 in document 180. The “0” values indicate that these terms are not present in these documents. The output of the “process document form data” operator is given to the set role operator.

Figure 18, is the word list that has been generated from the “Process data from Document” operator. Here we can see a word list containing all the different words in your document and their occurrence count next to it in the "Total Occurrences" column.

Result History | WordList (Process Documents from Data) | PerformanceVecto

Data

Word	Attribute Name	Total Occuren...	Document Occurences
flight	flight	980	791
thank	thank	459	443
get	get	294	276
cancel	cancel	238	218
hour	hour	215	204
delai	delai	211	202
time	time	199	183
custom	custom	188	179
servic	servic	187	184

Figure_18

Step_5: Performance

1. We used Cross-Validation operator to perform a cross-validate to estimate the statistical performance of a learning model. The performance of cross-validation is better as it split the dataset into training and testing independently using K-numbers of folds. We have used 10 folds. The Cross-Validation process has two subprocesses: Training Subprocess and Test Subprocess.
    - Algorithms: Our dataset is string therefore, we are going to use predictive algorithms like Decision Tree, Naïve Bayes, Random Forest, and K-NN. The training data is connected to the algorithm and the output is then connected to the "Apply Model".
    - Apply Model: This operator applies a model on the test data with unknown Sentiments to get a prediction on unseen data.
    - Performance: This operator is used for statistical performance evaluations of the model.

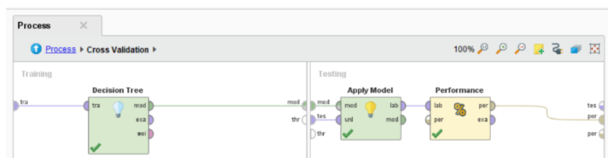


Figure_19: Decision Tree Algorithm

Result and Analysis

We applied four algorithms with and without pruning.

1. Decision Tree
2. Naïve Bayes
3. KNN
4. Random Forest

The results are noted down for evaluation. The following is the accuracy of all the algorithms: -

Naïve Bayes: - Confusion matrix – Accuracy – 65.38%

Table View Plot View

accuracy: 65.38% +/- 2.14% (mikro: 65.38%)

	true negative	true positive	class precision
pred. negative	1083	280	79.46%
pred. positive	735	834	53.15%
class recall	59.57%	74.87%	

Naïve Bayes (with pruning): - Confusion matrix – Accuracy – 69.00%

Table View Plot View

accuracy: 69.00% +/- 3.13% (mikro: 69.00%)

	true negative	true positive	class precision
pred. negative	1058	149	87.66%
pred. positive	760	965	55.94%
class recall	58.20%	86.62%	

KNN (10 folds): - Confusion matrix – Accuracy 39.02%

Table View Plot View

accuracy: 39.02% +/- 0.91% (mikro: 39.02%)

	true negative	true positive	class precision
pred. negative	44	14	75.86%
pred. positive	1774	1100	38.27%
class recall	2.42%	98.74%	

KNN (10 folds-With Pruning): - Confusion matrix – Accuracy 75.75%

Table View Plot View

accuracy: 75.75% +/- 1.36% (mikro: 75.75%)

	true negative	true positive	class precision
pred. negative	1733	626	73.46%
pred. positive	85	488	85.17%
class recall	95.32%	43.81%	

Random Forest (50 folds): - Confusion matrix – Accuracy 62.01%

Table View Plot View

accuracy: 62.01% +/- 0.95% (mikro: 62.01%)

	true negative	true positive	class precision
pred. negative	1818	1114	62.01%
pred. positive	0	0	0.00%
class recall	100.00%	0.00%	

Random Forest (50 folds-with pruning): - Confusion matrix – Accuracy 73.37%

Table View Plot View

accuracy: 73.37% +/- 4.11% (mikro: 73.36%)

	true negative	true positive	class precision
pred. negative	1785	748	70.47%
pred. positive	33	366	91.73%
class recall	98.18%	32.85%	

Decision Tree: - Confusion matrix – Accuracy 75.51%

Table View Plot View

accuracy: 75.51% +/- 3.29% (mikro: 75.51%)

	true negative	true positive	class precision
pred. negative	1747	647	72.97%
pred. positive	71	467	86.80%
class recall	96.09%	41.92%	

On applying all the four algorithms we can see that the decision tree classifier out-performs naïve Bayes', KNN and random forest.

Accuracy rate is the % of test set samples that are correctly classified by the model. To estimate the accuracy of the model, the known label of test data is compared with the classified result from the model. For the model that we have built, we can see that the accuracy is 75.51% and 1747 tweets that are actually negative are predicted as negative (True Negative-TN) and 467 tweets that are actually positive are predicted as positive (True Positive- TP). Similarly, 647 tweets that are actually positive are precited as negative (False Positive- FP) and 71 tweets that are actually negative are precited as positive (False Negative- FN).

The below figure 20 is the decision tree that is built by the operator. Based on the input data, all the negative words and the positive words are identified, and the tweet classification is done accordingly. In a decision tree by the process of recursive partitioning, the tree is created in such a way that the best predictors automatically bubbled to the top of the tree. The root node is the best predictor out of all the predictor variables that we used in this example set. If we look at the above tree, we can see that the word "thank" is the best predictor among all the attributes. If "thank" is greater than 0.302, than it is classified as positive. Similarly, we can interpret the rest of the decision tree diagram.

If we go to the detailed view (below figure 21), we can see the above information in a more detailed form. For example, for the word "thank", the TFIDF score is greater than 0.302, than it is classified as positive. Similarly, the classification is done for all the negative and positive words.

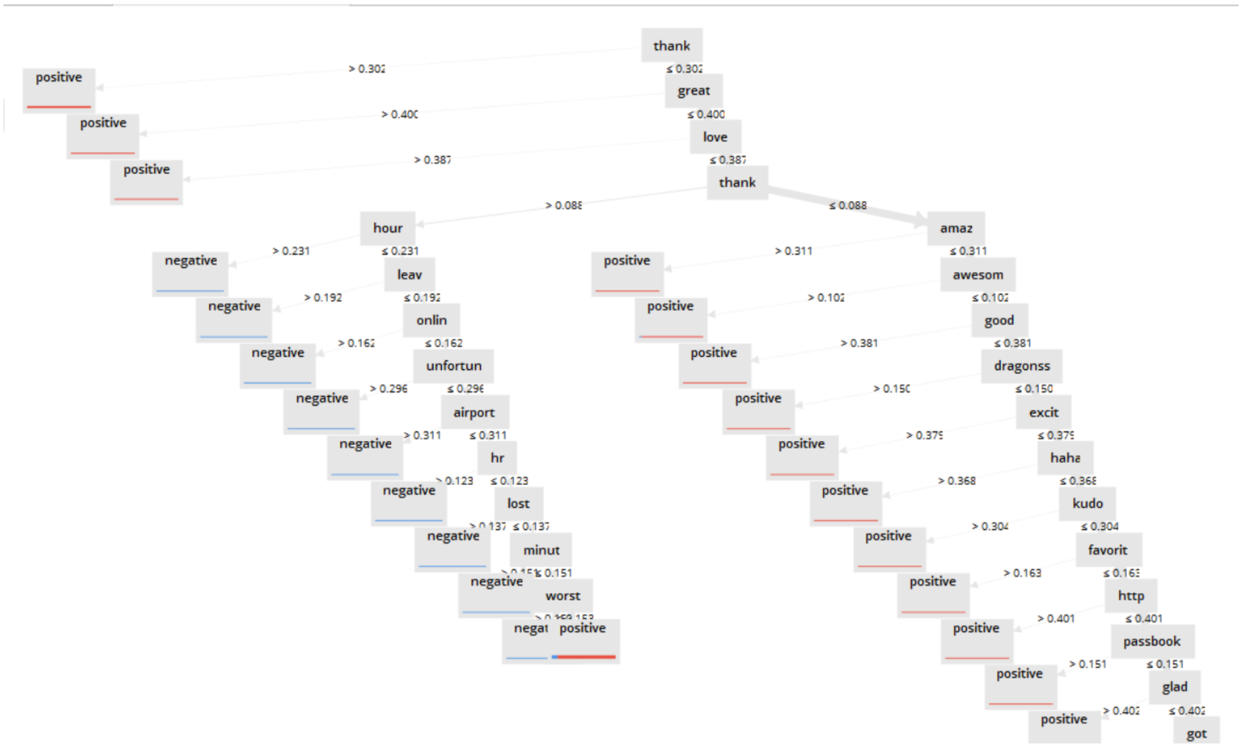


Figure_20: Decision Tree without pruning

Tree

```
thank > 0.302: positive {negative=0, positive=105}
thank ≤ 0.302
|   great > 0.400: positive {negative=0, positive=28}
|   great ≤ 0.400
|   |   love > 0.387: positive {negative=0, positive=20}
|   |   love ≤ 0.387
|   |   |   thank > 0.088
|   |   |   |   hour > 0.231: negative {negative=7, positive=0}
|   |   |   |   hour ≤ 0.231
|   |   |   |   |   leav > 0.192: negative {negative=3, positive=0}
|   |   |   |   |   leav ≤ 0.192
|   |   |   |   |   |   onlin > 0.162: negative {negative=3, positive=0}
|   |   |   |   |   |   onlin ≤ 0.162
|   |   |   |   |   |   |   unfortun > 0.296: negative {negative=3, positive=0}
|   |   |   |   |   |   |   unfortun ≤ 0.296
|   |   |   |   |   |   |   |   airport > 0.311: negative {negative=2, positive=0}
|   |   |   |   |   |   |   |   airport ≤ 0.311
|   |   |   |   |   |   |   |   |   hr > 0.123: negative {negative=2, positive=0}
|   |   |   |   |   |   |   |   |   hr ≤ 0.123
|   |   |   |   |   |   |   |   |   |   lost > 0.137: negative {negative=2, positive=0}
|   |   |   |   |   |   |   |   |   |   lost ≤ 0.137
|   |   |   |   |   |   |   |   |   |   |   minut > 0.151: negative {negative=2, positive=0}
|   |   |   |   |   |   |   |   |   |   |   minut ≤ 0.151
|   |   |   |   |   |   |   |   |   |   |   |   worst > 0.153: negative {negative=2, positive=0}
|   |   |   |   |   |   |   |   |   |   |   |   worst ≤ 0.153: positive {negative=24, positive=240}
|   |   |   thank ≤ 0.088
|   |   |   |   amaz > 0.311: positive {negative=0, positive=13}
|   |   |   |   amaz ≤ 0.311
|   |   |   |   |   awesom > 0.102: positive {negative=1, positive=27}
|   |   |   |   |   awesom ≤ 0.102
|   |   |   |   |   |   good > 0.381: positive {negative=0, positive=9}
|   |   |   |   |   |   good ≤ 0.381
|   |   |   |   |   |   |   dragonss > 0.150: positive {negative=0, positive=7}
|   |   |   |   |   |   |   dragonss ≤ 0.150
|   |   |   |   |   |   |   |   excit > 0.379: positive {negative=0, positive=7}
```

Figure_21: Decision Tree Detailed

We also conducted pruning and reapplied the decision tree algorithm in order to increase the performance of the model. Pruning was chosen to reduce the complexity of the final classifier, thereby improving the predictive accuracy of the model by the reduction of overfitting. In this case, we have used perceptual prune method with prune percent range of 3-30%.

Row No.	Sentiment	prediction(S...	confidence(negative)	confidence(positive)
1	negative	negative	0.766	0.234
2	negative	negative	0.766	0.234
3	negative	negative	0.766	0.234
4	negative	negative	0.766	0.234
5	negative	negative	0.766	0.234
6	negative	negative	0.766	0.234
7	positive	negative	0.766	0.234
8	negative	negative	0.766	0.234
9	positive	negative	1	0
10	negative	negative	0.766	0.234
11	positive	negative	0.766	0.234
12	negative	negative	0.766	0.234
13	positive	negative	0.766	0.234
14	negative	negative	0.766	0.234
15	positive	negative	0.766	0.234
16	negative	negative	0.766	0.234
17	positive	positive	0.083	0.917
18	positive	negative	0.766	0.234

Figure: Example Set of Performance

Decision Tree (with pruning): - Confusion Matrix- Accuracy 76.43

Table View Plot View

accuracy: 76.43% +/- 1.74% (mikro: 76.43%)

	true negative	true positive	class precision
pred. negative	1769	642	73.37%
pred. positive	49	472	90.60%
class recall	97.30%	42.37%	

We have seen almost 1% increase in the accuracy of the model. The true negative and true positive counts have also increased from 1747 to 1769 and 467 to 472 respectively. Also, the errors (FP and FN) have decreased. This shows that the motive of using the pruning was achieved as the decision tree looks pruned and clearer to understand (figure 21)

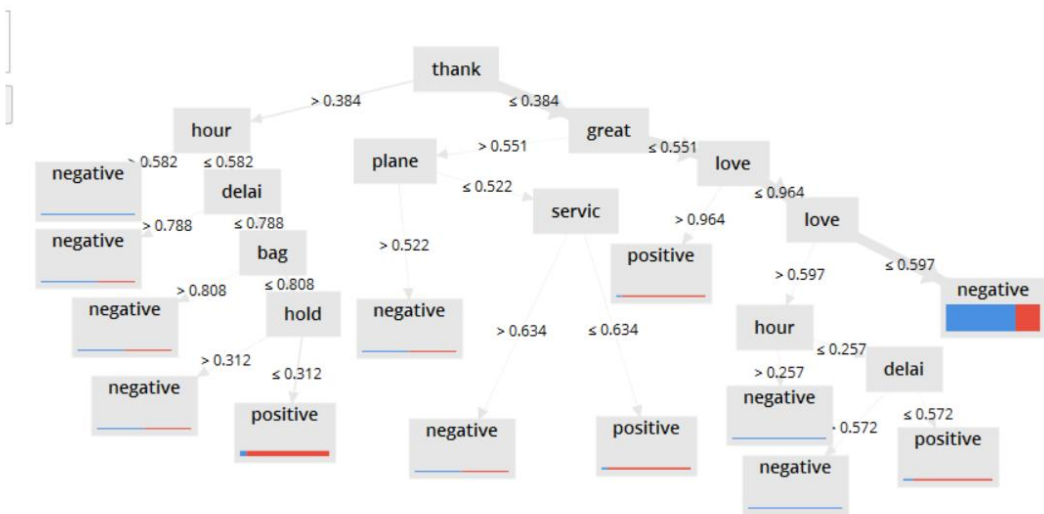


Figure_21: Decision Tree with Pruning

K-Means Clustering

We used k-means algorithm to perform clustering. Clustering groups examples together which are similar to each other. In our case, we tried to cluster the structured word vectors into two clusters.

1) To achieve this objective, we created a process by dragging and dropping the Airline data preparation (i.e. Retrieving data, subprocess 1, subprocess 2, process document from data).
2) We created a subprocess 3 and inside the subprocess, we selected the structured word data that came from process document and applied K-means Clustering.

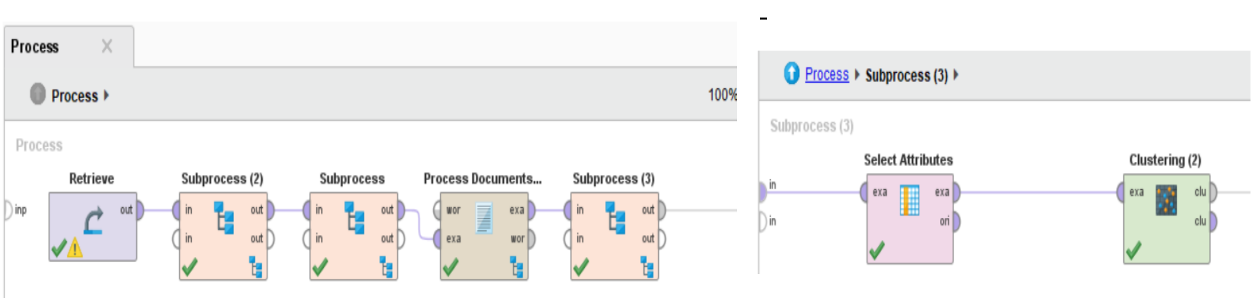


Figure_22: K-Means Clustering Model

Result and Analysis of Clustering

As we can see in the centroid table, we have two clusters. In cluster 0, predominately the word “thank”, “great”, “help” shows up whereas, in cluster 1, predominately the word “delay” and “cancel” shows up. We can infer from these findings that cluster 0 is primarily around tweets with positive sentiment, whereas cluster 1 is primarily around tweets with negative sentiments.

Result History | Cluster Model (Clustering (2))

Description | Folder View

Attribute	cluster_0 ↓
thank	0.852
flight	0.067
great	0.035
make	0.032
help	0.027

Cluster Model (Clustering

Cluster Model

```
Cluster 0: 376 items
Cluster 1: 2556 items
Total number of items: 2932
```

Figure_23: Centroid Table (Cluster_0 in descending order)

Result History | Cluster Model (Clustering (2))

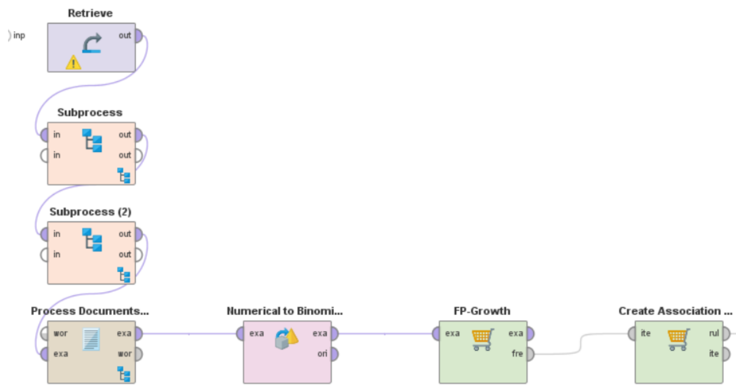
Description | Folder View

Attribute	cluster_0	cluster_1 ↓
flight	0.067	0.171
delai	0.017	0.057
hour	0.002	0.052
time	0.022	0.051
cancel	0.007	0.048

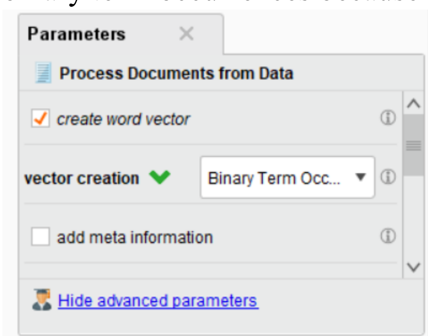
Figure_24: Centroid Table (Cluster_1 in descending order)

Association-Rule

This process would let us see the association among different words.



1) To achieve this objective, we created a new process and dragged and dropped the Airline data preparation (i.e. Retrieving data, subrocess1, subprocess 2, process document from data).
2) We made a small tweak in the "process document from data" by changing the vector creation tab from TF-IDF to binary term occurrence. We will use binary term occurrences because we can only use 1's or 0's for association analysis.



3) Drag and drop "Numerical to Binomial" operator. This operator changes the type of the selected numeric attributes to a binomial type. This operator is required because the FP-Growth operator that we would be using in the next steps requires all the terms to be in binomial form.
4) The change is clearly shown below (in Figure- 25 & Figure- 26)

Before: -

ExampleSet (2932 examples, 1 special attribute, 2698 regular attributes)

Row No.	text	abassinet	abl	absolut	absurd	abus	accept	access	accid
1	happi cancel ...	0	0	0	0	0	0	0	0
2	frustrat loooo...	0	0	0	0	0	0	0	0
3	updat text res...	0	0	0	0	0	0	0	0
4	agent on rud...	0	0	0	0	0	0	0	0
5	flight delai m...	0	0	0	0	0	0	0	0
6	thank comple...	0	0	0	0	0	0	0	0
7	love	0	0	0	0	0	0	0	0

Figure_25: Process Document from Data Result

After: -

ExampleSet (2932 examples, 1 special attribute, 2698 regular attributes)

Row No.	text	abassinet	abl	absolut	absurd	abus	accept	access	accid
1	happi cancel ...	false	false	false	false	false	false	false	false
2	frustrat loooo...	false	false	false	false	false	false	false	false
3	updat text res...	false	false	false	false	false	false	false	false
4	agent on rud...	false	false	false	false	false	false	false	false
5	flight delai m...	false	false	false	false	false	false	false	false
6	thank comple...	false	false	false	false	false	false	false	false
7	love	false	false	false	false	false	false	false	false

Figure_25: Process Document from Data Result after using Numerical to Binomial operator

5. Drag and drop “FP Growth” operator: - This operator efficiently calculates all frequent itemset that often appears together in the data from the given Example Set using the FP-tree data structure. We have set the min support to 0.01(1%) (see Figure - 4). Support is an indication of how frequently the items appear in the tweet data. We can change the support value if the model couldn’t find the min number of the dataset.

Parameters

FP-Growth

☑ find min number of itemsets

min number of items...	100
max number of retries	15
positive value	
min support	0.01
max items	3

In the below figure (Figure 26), we can interpret that flight has occurred 27% of the whole tweet data. Whereas customer and service together appeared 6% in the whole tweet data and flight, cancel and hold appeared 1% in the whole tweet dataset. (Figure- 27).

No. of Sets: 110
Total Max. Size: 2

Min. Size: 1
Max. Size: 2
Contains Item:

Update View

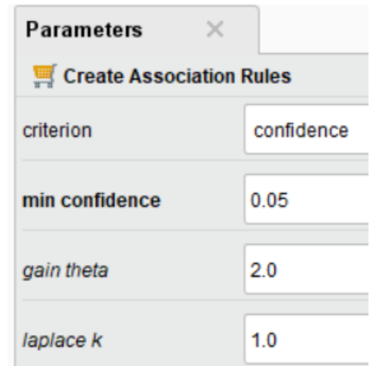
Size	Support	Item 1
1	0.270	flight
1	0.149	thank
1	0.074	cancel
1	0.070	hour
1	0.069	delai
1	0.063	servic
1	0.062	time
1	0.061	custom

Figure_26: FP-Growth Result

2	0.038	servic	custom	
2	0.011	fleet	fleek	
3	0.014	flight	cancel	hold
3	0.032	flight	cancel	flightl

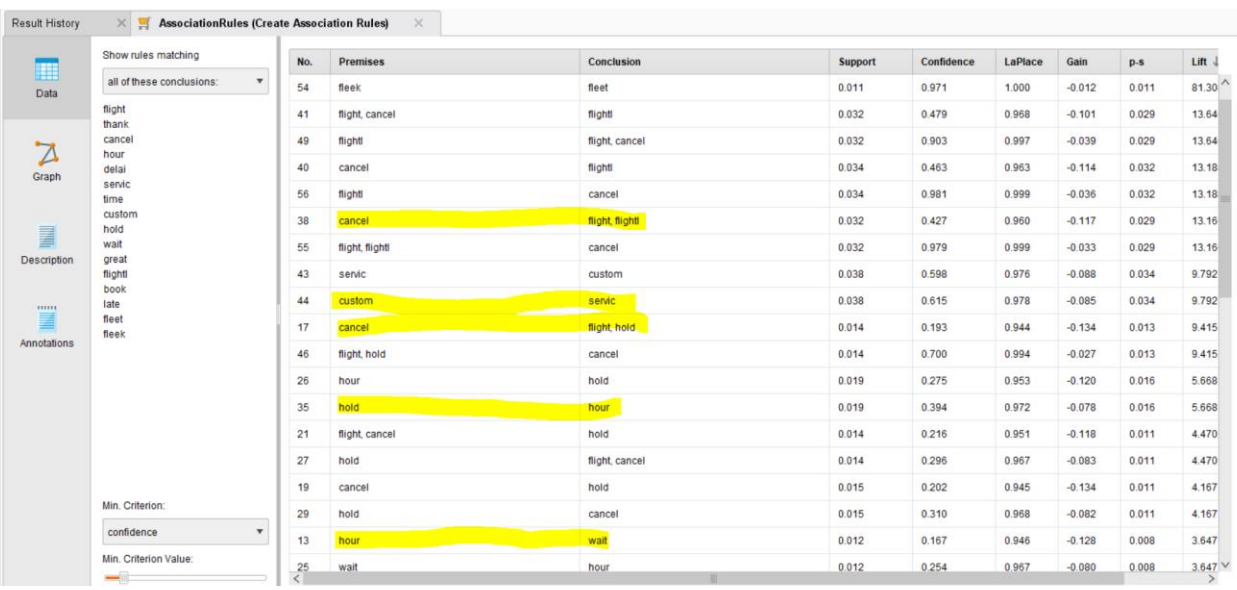
Figure_27: FP-Growth Result

6. Drag and Drop “Create association rule” operator: - This operator generates a set of association rules from the given set of the frequent itemset. The association rules are formed by analyzing data for frequent if/then patterns and using the criteria support and confidence to identify the most important associations. We have set the min confidence in the create association rule parameters to 0.05(5%). The confidence indicates the number of times the if/then statements have been found to be true.



Results and Analysis of Association-Rule

After running the model, we saw the following (Figure- 28) association rules that are common in our tweets: - Flight and cancel have been associated, hour and wait are associated with each other.



No.	Premises	Conclusion	Support	Confidence	LaPlace	Gain	p-s	Lift
54	fleek	fleet	0.011	0.971	1.000	-0.012	0.011	81.30
41	flight, cancel	flightl	0.032	0.479	0.968	-0.101	0.029	13.64
49	flightl	flight, cancel	0.032	0.903	0.997	-0.039	0.029	13.64
40	cancel	flightl	0.034	0.463	0.963	-0.114	0.032	13.18
56	flightl	cancel	0.034	0.981	0.999	-0.036	0.032	13.18
38	cancel	flight, flightl	0.032	0.427	0.960	-0.117	0.029	13.16
55	flight, flightl	cancel	0.032	0.979	0.999	-0.033	0.029	13.16
43	servic	custom	0.038	0.598	0.976	-0.088	0.034	9.792
44	custom	servic	0.038	0.615	0.978	-0.085	0.034	9.792
17	cancel	flight, hold	0.014	0.193	0.944	-0.134	0.013	9.415
46	flight, hold	cancel	0.014	0.700	0.994	-0.027	0.013	9.415
26	hour	hold	0.019	0.275	0.953	-0.120	0.016	5.668
35	hold	hour	0.019	0.394	0.972	-0.078	0.016	5.668
21	flight, cancel	hold	0.014	0.216	0.951	-0.118	0.011	4.470
27	hold	flight, cancel	0.014	0.296	0.967	-0.083	0.011	4.470
19	cancel	hold	0.015	0.202	0.945	-0.134	0.011	4.167
29	hold	cancel	0.015	0.310	0.968	-0.082	0.011	4.167
13	hour	wait	0.012	0.167	0.946	-0.128	0.008	3.647
25	wait	hour	0.012	0.254	0.967	-0.080	0.008	3.647

Figure_28: Association-Rule Result

If we try to visualize it using a graph and filter by each conclusion, we can see the results for all the associations with flight (Figure 29). Basically, the graph tells that the words mentioned on it often appear together in multiple documents. Similarly in Figure-30 and Figure-31.

Figure_29

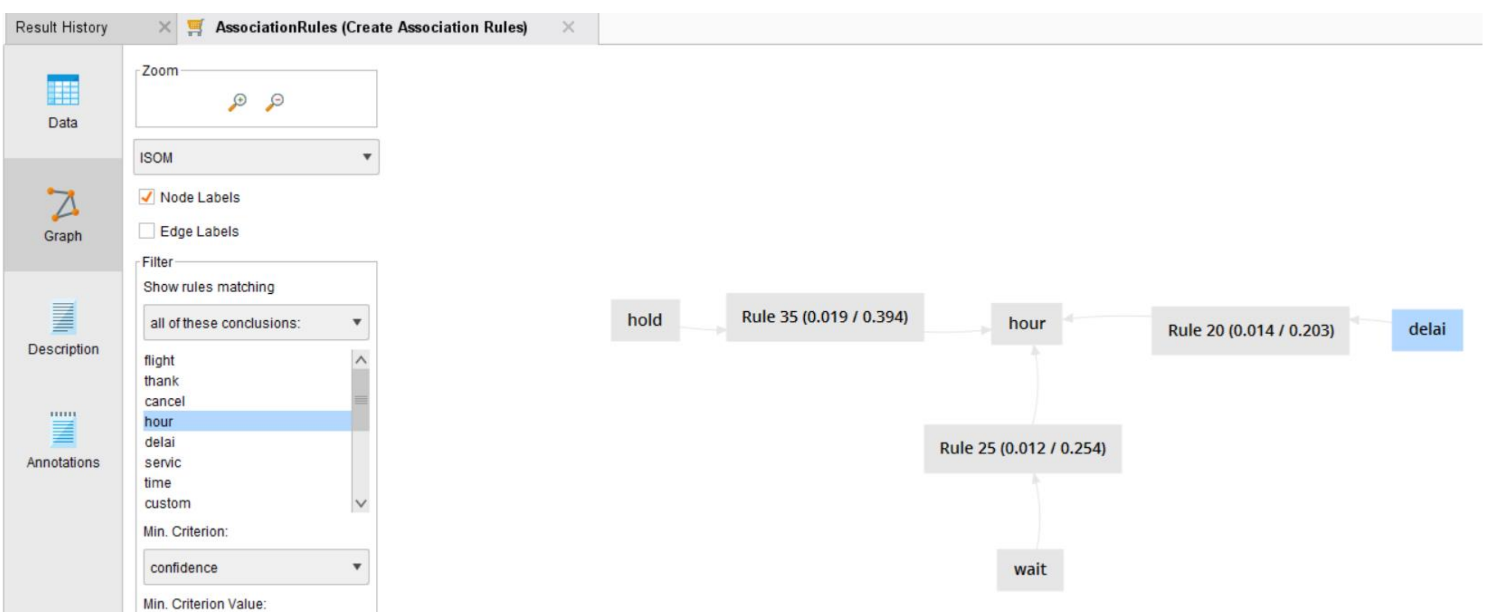


Figure_30

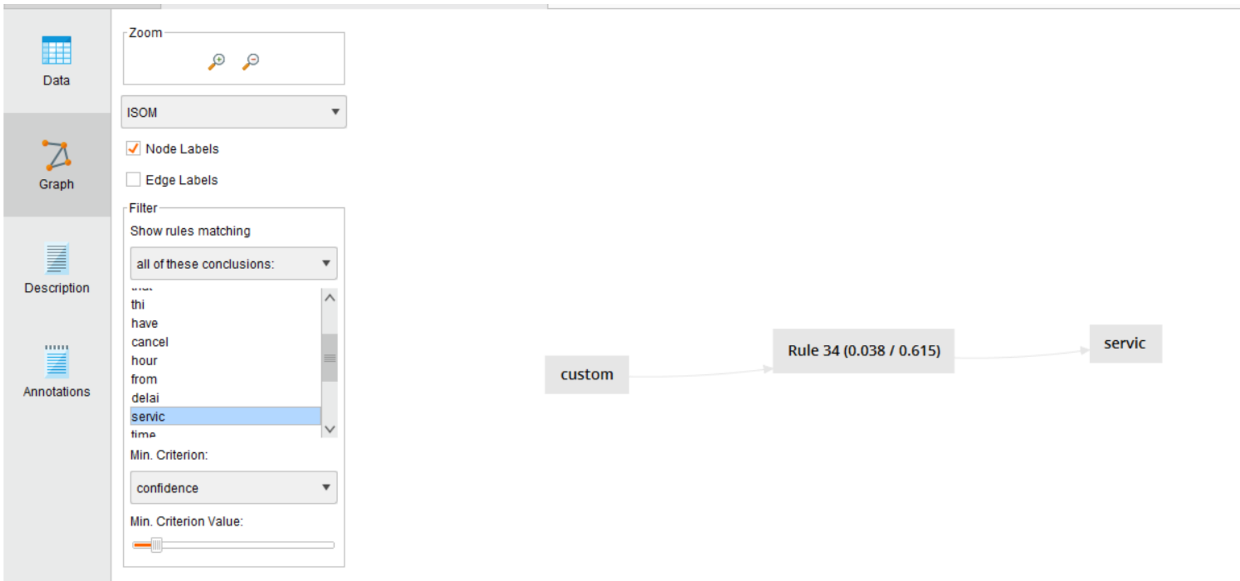

Data
Graph
Description
Annotations
Zoom
ISOM
Node Labels
Edge Labels
Filter
Show rules matching
all of these conclusions:
thi
have
cancel
hour
from
delai
servic
Min. Criterion:
confidence
Min. Criterion Value:
custom
Rule 34 (0.038 / 0.615)
servic

Figure_31

Conclusion

We got a good and balanced dataset as Kaggle had slightly reformatted the original version which was highly sophisticated. Since the success of predictive analytical models hugely depends on the quality of the data collected and the data preparation functions used to clean it, we spent 90% of our time cleaning and preparing the data for further analysis. Our efforts helped us in achieving a high-quality data and overall a better accuracy percentage. Based on professor's recommendation, we also referred to YouTube tutorials and peer-reviewed articles. This helped us understand the concept and necessity of a particular operator. For example, for "process document from data" operator to work, it needs text data. We knew we had to use nominal to text operator, but we were not sure where to use it. So, references to online videos and journals worked for us. Finally, we were pleased that our data process worked. Although we couldn't achieve the accuracy higher than 80%, decision tree model gave us the accuracy higher than 75%. We think one of the reasons behind less accuracy may be a result of the sarcasm that some of the tweets contain, which is hard for any model to analyze and interpret. We also benefited a lot by working as a team. Since data mining was new to both of us, we were each other's bouncing boards at times when we were in a fix. We had regular team meetings where our agenda was to review our work, address any concerns and make a plan for the next week.

Our initial plan was to retrieve live tweets from Twitter via Twitter API. We spent our initial week trying to build that connection and eventually succeeded. But the issue was the data limits that Twitter had on data retrieval. We could only get 400 tweets, and most of those were repeating. Given this scenario, we also started working on another dataset on WEKA as we wanted to keep a backup option in-case we couldn't make the text mining project work. This whole exercise consumed some time and energy. Also, we spent a lot of time selecting the dataset. Initially, we juggled with two datasets before finally landing on to the present dataset. The first dataset was on HI-B immigration. The issue with that dataset was that it was highly skewed as all the immigrants were posting negative tweets. Subsequently, we chose a new dataset on demonetization. The problem with this dataset was that most of the tweets were plain comments and opinions expressed by people. The number of tweets showing polarity (representing positive and negative sentiments) was less. We were also limited in our choice of algorithms. Since we were doing text mining, we couldn't use algorithms like linear and logistic regression models. Also cleaning text data was highly challenging initially. For example, as part of data cleaning, we replaced all non-text data like (@, #,?,!, HTTP://), but it was still showing in the text data once we ran the process. After spending some time on the YouTube, we realized that we have to use "Filter tokens (By Content) to remove all these characters. These initial hurdles took more time than necessary.

Next time we'll invest more time in studying the data and getting familiar with its features, before rushing on to incorporate it in the data mining tool. In future, we can also work on feature generation component of data mining. We can incorporate RapidMiner Wordnet Dictionary for Synonyms to improve our process and increase our model's accuracy. Moreover, we can explore even richer linguistic analysis like parsing, semantic analysis, and part-of-speech tags. Our aim was to generate the best prediction model that can predict sentiments and help airlines to get a quick overview of their reputation. We could take a step ahead by automatically filtering the tweets based on its content for the airlines that requires urgent attention. This way airlines can respond to urgent issues and provide better customer service.

The use of word vectors as features for representing tweets seems to be effective from our project. Based on the accuracy matrix it is logical to assume that the word list we used contains a substantial number of typical sentiment words. Hence, we believe that the availability of a powerful word list is a crucial component of our approach to be successful.

References

Ertek, G., Tapucu, D., & Arin, I. (2013). Text mining with rapidminer. *RapidMiner: Data Mining Use Cases and Business Analytics Applications*, 241.

Giachanou, A., & Crestani, F. (2016). Like it or not: A survey of twitter sentiment analysis methods. *ACM Computing Surveys (CSUR)*, *49*(2), 28.

Hu, G., Bhargava, P., Fuhrmann, S., Ellinger, S., & Spasojevic, N. (2017). Analyzing users' sentiment towards popular consumer industries and brands on Twitter. *arXiv preprint arXiv:1709.07434*.

Kharde, V., & Sonawane, P. (2016). Sentiment analysis of twitter data: A survey of techniques. *arXiv preprint arXiv:1601.06971*.

Pak, A., & Paroubek, P. (2010, May). Twitter as a corpus for sentiment analysis and opinion mining. In *LREc* (Vol. 10, No. 2010).

Thakkar, H., & Patel, D. (2015). Approaches for sentiment analysis on twitter: A state-of-art study. *arXiv preprint arXiv:1512.01043*.

Wills, R. K. (2012). Efficient Sentiment Analysis of Feeds for Rapid User Information Gain.

Yuan, P., Zhong, Y., & Huang, J. Sentiment Classification and Opinion Mining on Airline Reviews.

YouTube Videos Links: https://www.youtube.com/watch?v=kFGronMuchU

https://www.youtube.com/watch?v=K7AgCas0gJ0

https://www.youtube.com/watch?v=kq61oFXD4YI