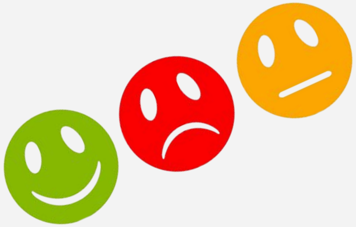


Sentiment Analysis in Unstructured text data

Presented By:
Priyanka Boppana
Gayatri Kakumanu
Prathima Paruchuri
Chaoyi Huang

Introduction

Sentiment Analysis

Identify and categorize the opinions expressed in a piece of text

- Positive
- Negative
- Neutral

The Sentiment Analysis uses two approaches

- Lexicon Based
- Machine Learning

Problem Definition:

The most common way for people to do sentiment analysis today is **Lexicon-based method** - by using word dictionary that contains thousands of positive, negative and neutral words to give sentiment score in different texts. This dictionary was generated manually by people, as well as the tag on each words.

When we applied this method in unstructured text data, the accuracy of sentiment analysis drop down significantly due to the simple parameters

Machine Learning

Definition: Machine learning is the semi-automated extraction of knowledge from data.

Main categories of machine learning:

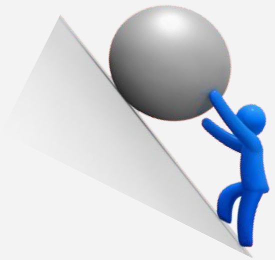
1. **Supervised Learning**- Making predictions using data.
2. **Unsupervised Learning** - Extracting structure from data.

Objective:

- Is to find out which method is more appropriate for a twitter based unstructured text data between Lexicon-based analysis and some machine learning methods.

- Is to improve the accuracy of unstructured data by combining some methods is the goal of our project.

Challenges



- **Tweets are highly Unstructured**

@Listen to #Attention on @AppleMusic's Global Pop playlist! http://apple.co/28M5kC2

- **Lexical Variation**

@USAirways @AmericanAir #OneHourOnHold,hatttttteeeeee it.

Languages:

- R language: Includes all tools necessary for web scraping, familiarity and direct analysis of data.

Proposed Technique:

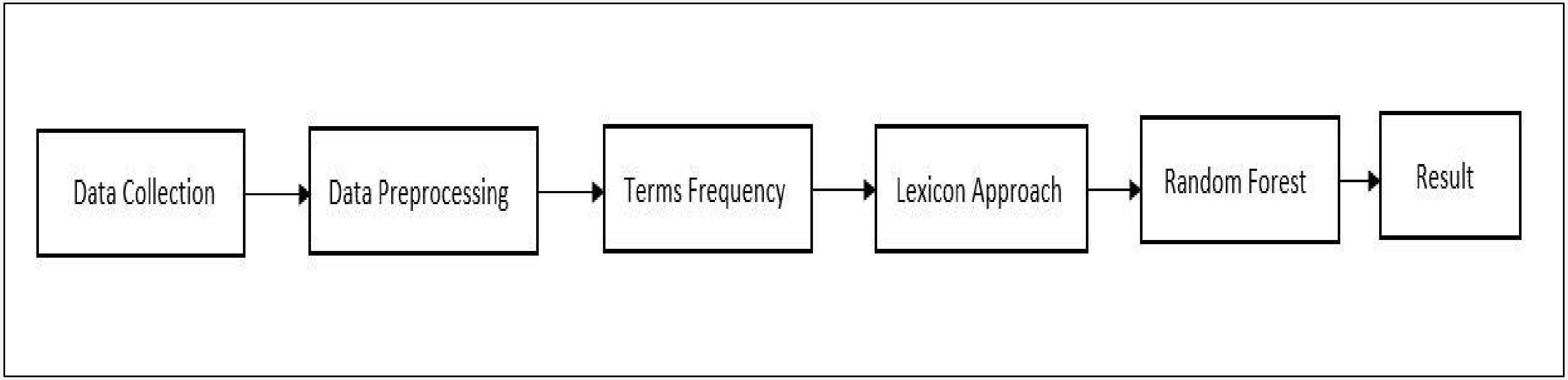


Figure: Systematic procedure for Predicting the data

Dataset

American Airline tweets positive sentiment only

- contains 336 tweets

IMBD movie review

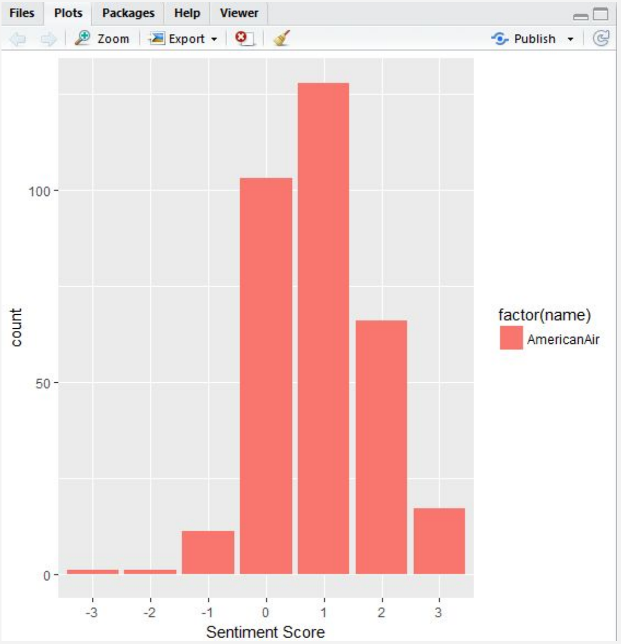
- Labeled training set (25,000 rows containing an id, sentiment and text for each review)
- Unlabeled training set (50,000 rows containing an id and text for each review)
- Test set (25,000 rows containing an id and text for each review)

Data-preprocessing

1. Convert all instances to lower cases
2. Removes urls
3. Removes punctuations
4. Removes numbers
5. Removes stopwords
6. Removes extra white spaces

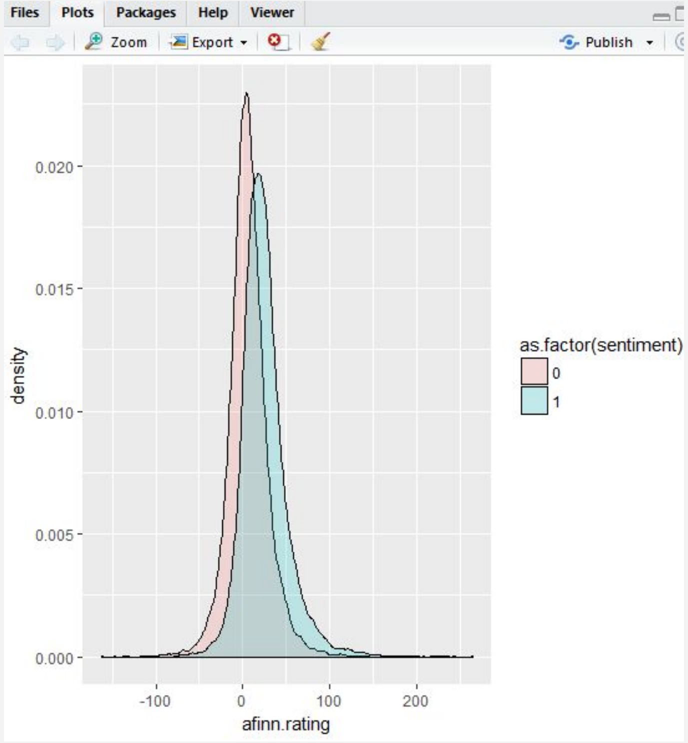
Lexicon-based approach

- **Dataset:** Tweets dataset contains positive sentiments only.
- **Dictionary:** AFINN contains 2700 positive words and 4900 negative words
- **Accuracy:** 73%
- **Pro:** Easy to use
- **Con:** Huge overlap between two classes.



Lexicon-based approach

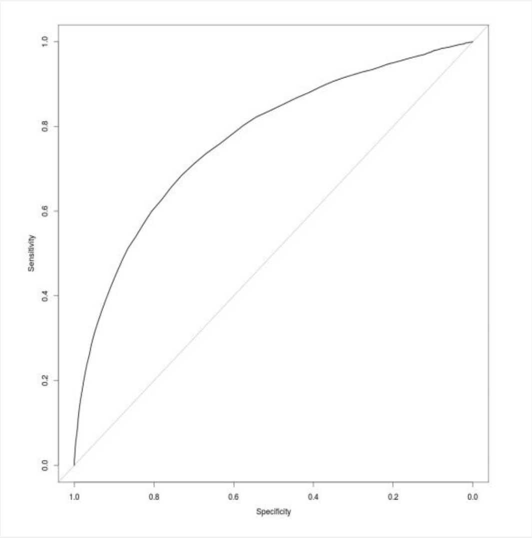
- **Dataset:** IMBD movie review
- **Dictionary:** AFINN
- **Accuracy:** 71%



Naive Bayes and Unsupervised Learning

Approach: Naive Bayes

Accuracy: AUC = 0.77516



Approach: Random Forest

Accuracy: AUC = 0.7858

Solution

1. Building a Term frequency Matrix from Corpus (75000*213398)

2. Remove all the stop words and the words occur very infrequently

3. Now we have a more manageable 9,799 columns

```
> head(colnames(tf))
[1] "actual"     "alone"      "also"       "another"    "anyway"     "attention"
```

Contd..

4. Create a word frequency data frame

```
word freq
5779 movie 125307
3375 film 113054
6132 one 77447
5150 like 59147
4847 just 53132
3826 good 43279
```

Contd..

5. Now we are building features on words that occur more often in positive review than in negative reviews.

	word	freq.x	freq.y	diff
1235	movie	23668	18139	5529
146	bad	7089	1830	5259
826	great	2601	6294	3693
1008	just	10535	7098	3437
604	even	7604	4899	2705
2115	worst	2436	246	2190

Contd..

6. We use NDSI, which is the difference of frequencies normalized by their sum. NDSI values are between 0 and 1 with higher values indicating greater correlation with sentiment.

$$NDSI(t) = |n(t|0) - n(t|1)|/n(t|0) + n(t|1)$$

7. We need to penalize infrequent words

$$NDSI(t) = |n(t|0) - n(t|1)|/n(t|0) + n(t|1) + 2\alpha$$

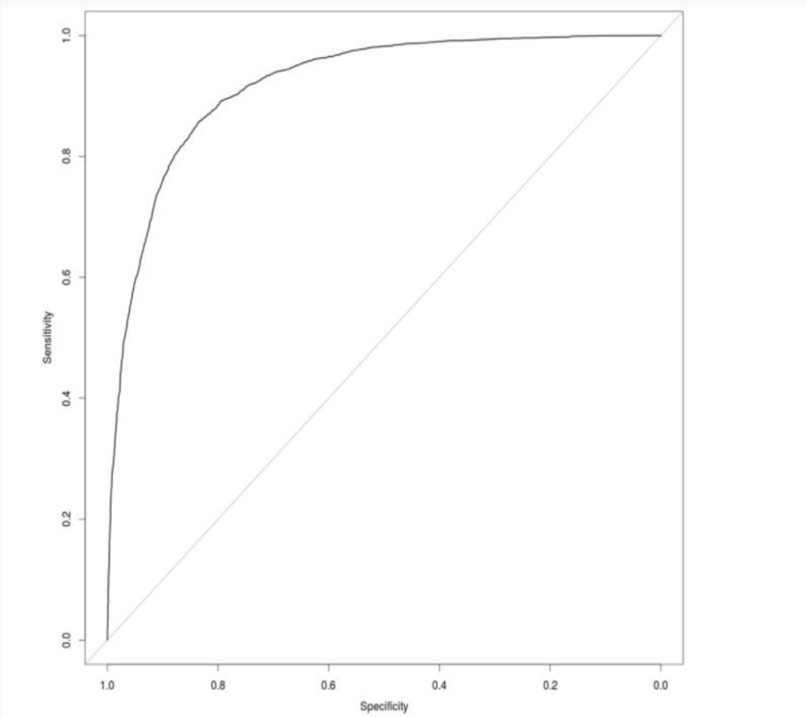
```
alpha <- 2**7

freq.all$ndsi <- abs(freq.all$freq.x - freq.all$freq.y)/
                         (freq.all$freq.x +
                          freq.all$freq.y +
                          2*alpha)
```

	word	freq.x	freq.y	diff	ndsi
2115	worst	2436	246	2190	0.7454050
2048	waste	1351	94	1257	0.7389771
1411	poorly	620	0	620	0.7077626
1040	lame	618	0	618	0.7070938
141	awful	1441	159	1282	0.6907328
1187	mess	498	0	498	0.6604775

Contd..

8. Apply our unsupervised machine learning (Random forest)

AUC = 0.9191



Conclusion and Future work

Pros: Higher accuracy, work on large dataset, matrix is easy to create

Con: Does not consider word meanings and similarities

Future:

Adding additional predictors to improve our predictions such as topic modeling and Clustering.