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,hattttttteeeeeeit.
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:** $\text{AUC} = 0.77516$

**Approach:** Random Forest

**Accuracy:** $\text{AUC} = 0.7858$
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

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

4. Create a word frequency data frame

<table>
<thead>
<tr>
<th>word freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>5779 movie 125307</td>
</tr>
<tr>
<td>3375 film 113054</td>
</tr>
<tr>
<td>6132 one 77447</td>
</tr>
<tr>
<td>5150 like 59147</td>
</tr>
<tr>
<td>4847 just 53132</td>
</tr>
<tr>
<td>3826 good 43279</td>
</tr>
</tbody>
</table>
Contd..

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

<table>
<thead>
<tr>
<th>word</th>
<th>freq.x</th>
<th>freq.y</th>
<th>diff</th>
</tr>
</thead>
<tbody>
<tr>
<td>1235</td>
<td>23668</td>
<td>18139</td>
<td>5529</td>
</tr>
<tr>
<td>146</td>
<td>7089</td>
<td>1830</td>
<td>5259</td>
</tr>
<tr>
<td>826</td>
<td>2601</td>
<td>6294</td>
<td>3693</td>
</tr>
<tr>
<td>1008</td>
<td>10535</td>
<td>7098</td>
<td>3437</td>
</tr>
<tr>
<td>604</td>
<td>7604</td>
<td>4899</td>
<td>2705</td>
</tr>
<tr>
<td>2115</td>
<td>2436</td>
<td>246</td>
<td>2190</td>
</tr>
</tbody>
</table>
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.

\[ \text{NDSI}(t) = \frac{|n(t|0) - n(t|1)|}{n(t|0) + n(t|1)} \]

7. We need to penalize infrequent words

\[ \text{NDSI}(t) = \frac{|n(t|0) - n(t|1)|}{n(t|0) + n(t|1) + 2\alpha} \]

\[
\text{alpha} <- 2^{**7} \\
\text{freq.all$ndsi} <- \frac{abs(freq.all$freq.x - freq.all$freq.y)}{(freq.all$freq.x + freq.all$freq.y + 2*alpha)}
\]

<table>
<thead>
<tr>
<th>word</th>
<th>freq.x</th>
<th>freq.y</th>
<th>diff</th>
<th>ndsi</th>
</tr>
</thead>
<tbody>
<tr>
<td>2115</td>
<td>2436</td>
<td>246</td>
<td>2190</td>
<td>0.7454050</td>
</tr>
<tr>
<td>2048</td>
<td>1351</td>
<td>94</td>
<td>1257</td>
<td>0.7389771</td>
</tr>
<tr>
<td>1411</td>
<td>620</td>
<td>0</td>
<td>620</td>
<td>0.7077626</td>
</tr>
<tr>
<td>1040</td>
<td>618</td>
<td>0</td>
<td>618</td>
<td>0.7070938</td>
</tr>
<tr>
<td>141</td>
<td>1441</td>
<td>159</td>
<td>1282</td>
<td>0.6907328</td>
</tr>
<tr>
<td>1187</td>
<td>498</td>
<td>0</td>
<td>498</td>
<td>0.6604775</td>
</tr>
</tbody>
</table>
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.