**AIDS Lab**

**EXPERIMENT NO. 4**

**Aim**: To build an adaptive and contextual Cognitive based Customer service application/Insurance/Healthcare Application/Smarter Cities/Government etc.

**Theory**:

Sentiment analysis (opinion mining) is a text mining technique that uses machine learning and natural language processing (nlp) to automatically analyze text for the sentiment of the writer (positive, negative, neutral, and beyond).

The overall purpose of text mining is to derive high-quality information and actionable insights from text, allowing businesses to make informed decisions. Powerful machine learning algorithms can easily recognize statements as Positive, Negative, or Neutral. And you can get even more granular results when you put aspect-based sentiment analysis into practice.

Aspect-based sentiment analysis takes it one step further, by organizing text like customer feedback or product reviews, first by category (Features, Shipping, Customer Service, etc.), and then mining text for sentiment so you can see which categories are positive or negative.

Sentiment analysis is what you might call a long-tail problem. Lots of varying scenarios and subtleties. Such problems are often best described by examples.

First, let’s see some easy positives.

*Amazing customer service.*

*Love it.*

*Good price.*

*Next, some positives and negatives are a bit harder to discriminate against.*

Positives:

*What is not to like about this product?*

*Not bad.*

*Not an issue.*

*Not buggy.*

Negatives:

*Not happy.*

*Not user-friendly.*

*Not good.*

Definitely not positive:

*Is it any good?*

The positives in the above list are not the strongest ones. That said, they are especially good for training ML algorithms to make key distinctions, as we definitely don’t want these positives to be predicted as negatives.

Positives:

*Low price.*

Negatives:

*Low quality.*

These instances are especially good for training ML algorithms to make key distinctions. There are many use cases. Here are some of the main specific ones.

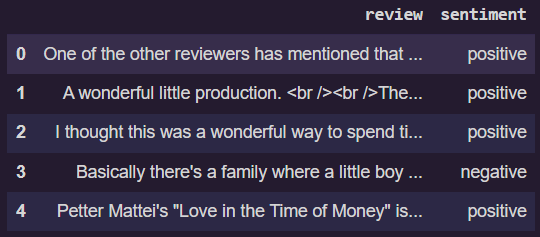
1. Discover negative reviews of your product or service. On blog posts or eCommerce sites or social media. More broadly anywhere on the web.
2. Aggregate sentiment on financial instruments. Such as specific stocks. What is the recent market sentiment on stock xyz? Also, aspect-based variants. Such as according to analysts at financial company xyz, stock abc is likely to grow 20% in the coming year. Discerning who’s opinion it is provides more information, which may be used to assess credibility or lack thereof.
3. Identify which components of your product or service are people complaining about? Especially strongly. For prioritizing tactical or long-term improvements.
4. Track changes to customer sentiment over time for a specific product or service (or a line of these). To check if things have been getting better.
5. Track shifting opinions of politicians over time. Individuals or groups such as political parties. News media love to do this. To fuel nagging questions such as you said that then but now this.

**Code and Output**:

Manually creating a model for text sentiment Analysis (Bag of Words Vectorization).

| from sklearn.feature\_extraction.text import CountVectorizer from nltk.tokenize import RegexpTokenizer !unzip archive.zip import pandas as pd data = pd.read\_csv('IMDB Dataset.csv') |
| --- |

| data.head() |
| --- |



| token = RegexpTokenizer(r'[a-zA-Z0-9]+') cv = CountVectorizer(stop\_words='english', ngram\_range=(1,1), tokenizer=token.tokenize) text\_counts = cv.fit\_transform(data['review']) from sklearn.model\_selection import train\_test\_split X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(text\_counts, data['sentiment'], test\_size=0.25, random\_state=5) from sklearn.naive\_bayes import MultinomialNB MNB = MultinomialNB() MNB.fit(X\_train, Y\_train) |
| --- |



| from sklearn import metrics predicted = MNB.predict(X\_test) accuracy\_score = metrics.accuracy\_score(predicted, Y\_test) print("Accuracy Score: ",accuracy\_score) |
| --- |



Using TextBlob, polarity determines the sentiment of the text. Its values lie in [-1,1] where -1 denotes a highly negative sentiment and 1 denotes a highly positive sentiment.

Subjectivity determines whether a text input is factual information or a personal opinion. Its value lies between [0,1] where a value closer to 0 denotes a piece of factual information and a value closer to 1 denotes a personal opinion.

| from textblob import TextBlob text\_1 = "AIDS is a good subject" text\_2 = "It was not a good idea." p\_1 = TextBlob(text\_1).sentiment.polarity p\_2 = TextBlob(text\_2).sentiment.polarity print("Polarity of Text 1 is", p\_1) print("Polarity of Text 2 is", p\_2) |
| --- |



This tells us that the first sentence is highly positive and the second one is highly negative.

| s\_1 = TextBlob(text\_1).sentiment.subjectivity s\_2 = TextBlob(text\_2).sentiment.subjectivity print("Subjectivity of Text 1 is", s\_1) print("Subjectivity of Text 2 is", s\_2) |
| --- |



The subjectivity tells us that both the sentences are personal opinions.

| text\_3 = "Mumbai is in Maharashtra" blob1 = TextBlob(text\_3) blob1.sentiment |
| --- |



Here the sentence is neutral and since it is factual information the subjectivity is zero.

Now using VADER (Valence Aware Dictionary and sEntiment Reasoner)

| !pip install vaderSentiment --quiet from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer sentiment = SentimentIntensityAnalyzer() sent\_1 = sentiment.polarity\_scores(text\_1) sent\_2 = sentiment.polarity\_scores(text\_2) print("Sentiment of text 1:", sent\_1) print("Sentiment of text 2:", sent\_2) |
| --- |



Finally using Transformer-Based Models.

| !pip install transformers --quiet import transformers from transformers import pipeline sentiment\_pipeline = pipeline("sentiment-analysis") data = ["The weather was awesome.", "My head is paining"] sentiment\_pipeline(data) |
| --- |



**Conclusion**:

Thus we studied an overview of what is Text Sentiment analysis and implemented an adaptive and contextual Cognitive based Customer service application.