REAL-TIME SENTIMENT ANALYSIS

Objective

The aim is to classifying emotions present in the text documents by applying various NLP techniques such as Tokenization, Lemmatization, POS tagging and many more.

Problem Statement

To understand and classify polarity of sentences within text documents at sentence-level analysis using Natural Language Processing.

Positive

Brilliant Work!
Good Job!

Neutral

Job is Doable.
Plan sounds Okay.

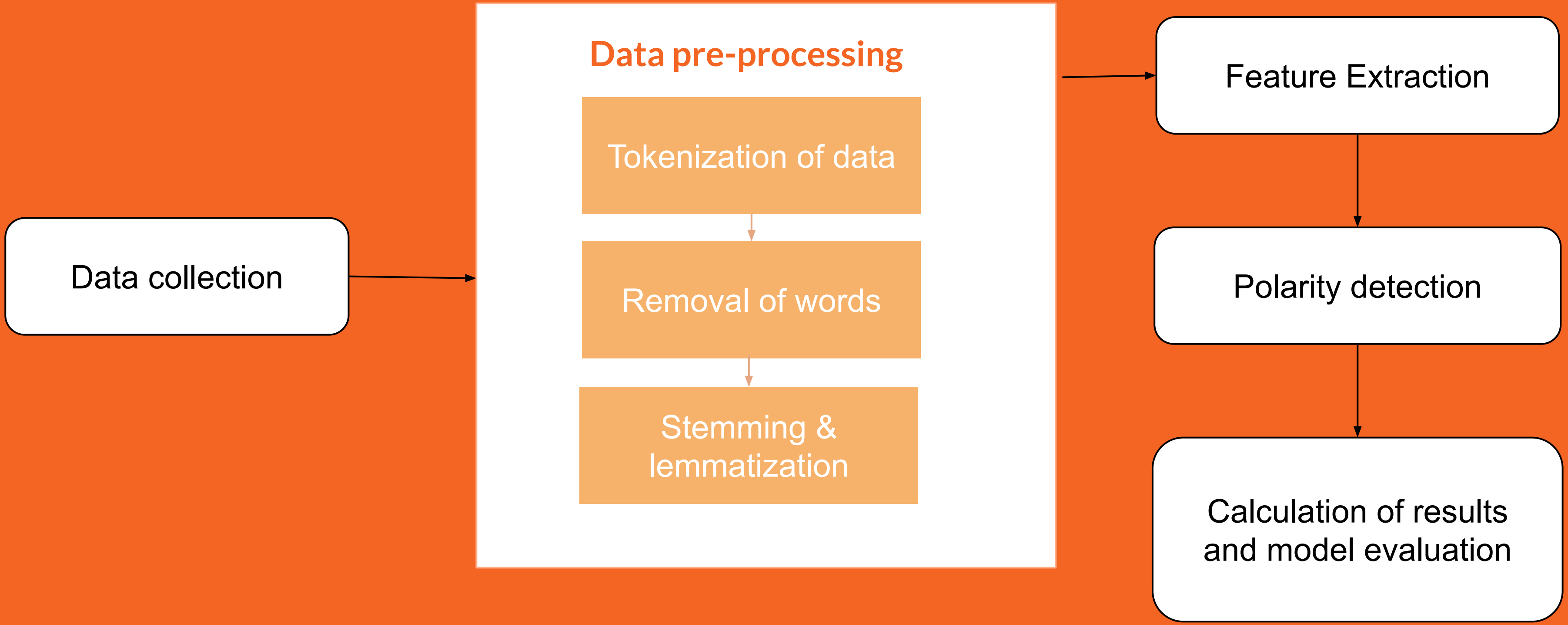
Negative

Cake tastes Bitter.
I hate walking.

What is Sentiment Analysis?

Sentiment Analysis (emotion AI) is a sub-field of NLP that tries to identify & extract opinions within a given text across blogs, reviews, social media, forums, news etc. It's a Natural Language Processing algorithm that gives you a general idea about the positive, neutral, and negative sentiment of texts.

Architecture



Related Work and Novelty

- There has been a steady undercurrent of interest for quite a while. For example, determining whether a review is positive or negative.[1]
- A number of researchers have explored learning words and phrases with prior positive or negative polarity.[2]
- Another work[3] where the researchers classify the contextual polarity of sentiment expressions and use manually developed patterns to classify polarity.

In contrast, we begin with a lexicon of words with established prior polarities and identify the contextual polarity of phrases and sentences in which instances of those words appear in the corpus. In addition to this, our system assign one sentiment per sentence as well as it assigns contextual polarity to individual expressions because usually sentences often contain more than one sentiment expression.

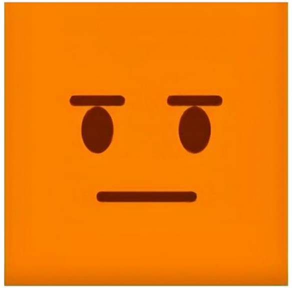
Data pre-processing

1. **Tokenization of data**: In this process, after the data is retrieved from the dataset. The data which is taken is in the form of sentences and phrases. Now, these sentences and phrases are tokenized into words, so that it is easily understandable.
2. **Removal of stop words**: Stop words are the words that the search engine has programmed to ignore when both indexing and retrieving of entries. Articles are the best examples of stop words.
3. **Stemming and Lemmatization**: Stemming and lemmatization is the process of converting the words into their root words so that they can be analyzed as a single item.
4. **P-O-S tagging**: Parts-of-Speech tagging is optional in sentiment polarity detection. It helps us to classify the words into a verb, adverb, adjective, noun etc. These words are then mapped to sentiment dictionary.

Sentiment Analysis

Happy

Sad

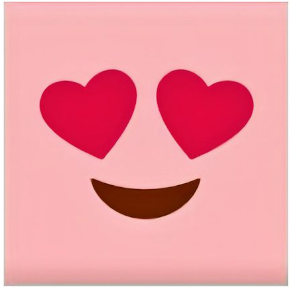


Angry

Disappointed

Surprised

Proud



In Love

Scared

```
for l in y_val:

    if l == "joy":
        int_y_val.append(0)
    if l == "sadness":
        int_y_val.append(1)
    if l == "anger":
        int_y_val.append(2)
    if l == "fear":
        int_y_val.append(3)
    if l == "love":
        int_y_val.append(4)
    if l == "surprise":
        int_y_val.append(5)

int_y_train,int_y_test,int_y_val = np.array(int_y_train),np.array(int_y_test),np.array(int_y_val)

from sklearn import preprocessing
from keras.utils import np_utils

le = preprocessing.LabelEncoder()
le.fit(int_y_train)

encoded_y_train = le.transform(int_y_train)
encoded_y_test = le.transform(int_y_test)
encoded_y_val = le.transform(int_y_val)

encoded_y_train = np_utils.to_categorical(encoded_y_train)
encoded_y_test = np_utils.to_categorical(encoded_y_test)
encoded_y_val = np_utils.to_categorical(encoded_y_val)

print(encoded_y_train)

print(int_y_train[:10])
```

Data Preparation

```
import nltk
nltk.download('stopwords')
from tensorflow.python.keras.preprocessing.text import Tokenizer
from tensorflow.python.keras.preprocessing.sequence import pad_sequences
from nltk.corpus import stopwords
```

```
stopwords = stopwords.words('english')
stopwords[:5]
```

```
['i', 'me', 'my', 'myself', 'we']
```

```
x_train_cl = []
x_test_cl = []
x_val_cl = []


# Deleting stopwords
for text in x_train:

    text = text.split()
    text = [word for word in text if word not in stopwords]
    text = " ".join(text)
    x_train_cl.append(text)

for text in x_test:

    text = text.split()
    text = [word for word in text if word not in stopwords]
    text = " ".join(text)
    x_test_cl.append(text)
```

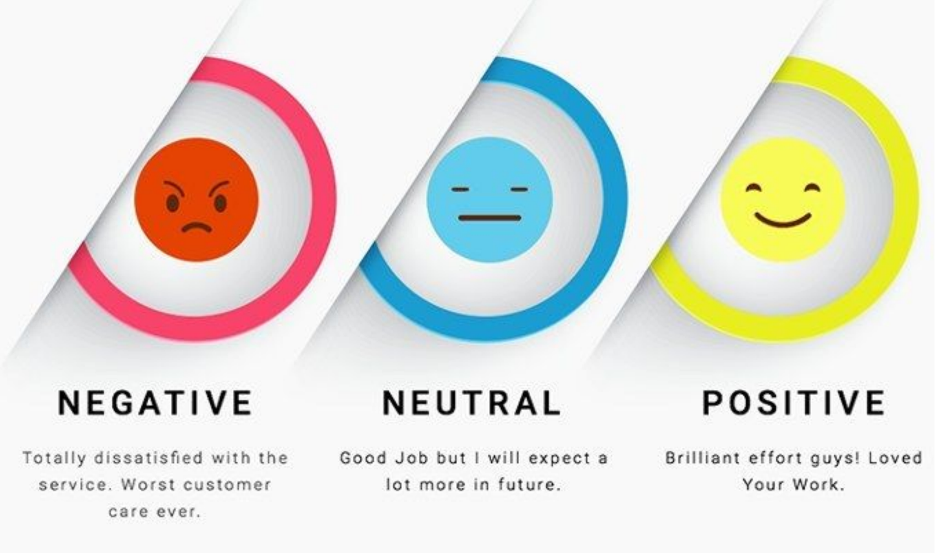
Tokenization

```
from tensorflow.python.keras.models import Sequential
from tensorflow.python.keras.layers import Dense,CuDNNGRU,Embedding,Bidirectional


model = Sequential()

model.add(Embedding(input_dim=2000
                    ,output_dim=100
                    ,input_length=20))

model.add(Bidirectional(CuDNNGRU(units=16,return_sequences=True)))

model.add(Bidirectional(CuDNNGRU(units=8)))

model.add(Dense(6,activation="softmax"))

model.compile(loss="categorical_crossentropy",optimizer="rmsprop",metrics=["accuracy"])

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 20, 100)           200000
_________________________________________________________________
bidirectional (Bidirectional (None, 20, 32)            11328
_________________________________________________________________
bidirectional_1 (Bidirection (None, 16)                2016
_________________________________________________________________
dense (Dense)                (None, 6)                 102
=================================================================
Total params: 213,446
Trainable params: 213,446
Non-trainable params: 0
_________________________________________________________________
```

Model and Prediction

Applications



- ❏ Social Media Monitoring
- ❏ Customer Support and feedback
- ❏ Brand monitoring and reputation management
- ❏ Product analysis
- ❏ Market and competitor research

Conclusion

Thus, polarity detection on the text document is performed and different emotions are observed and classified into positive, negative and neutral by using NLTK.

References

1. Riloff and J. Wiebe. 2003. Learning extraction patterns for subjective expressions. In *EMNLP-2003*.
2. Takamura, Hiroya & Inui, Takashi & Okumura, Manabu. (2005). *Extracting emotional polarity of words using spin model.*
3. J. Yi, T. Nasukawa, R. Bunescu and W. Niblack, "Sentiment analyzer: extracting sentiments about a given topic using natural language processing techniques," *Third IEEE International Conference on Data Mining,* Melbourne, FL, USA, 2003, pp. 427-434, doi: 10.1109/ICDM.2003.1250949.

Thank You

Akshaya A - CB.EN.U4CSE17401

Divya Rathi - CB.EN.U4CSE17416

Gayathri E - CB.EN.U4CSE17420

Kaviya S - CB.EN.U4CSE17429