Sentiment analysis for marketing

Phase 3 submission document

Project tittle:sendiment analysis for marketing

Phase 3: development part 1

Topic: In this section begin building your project by loading and preprocessing the dataset.


Text Input
Tokenization
Stop Word Filtering
Negation Handling
Stemming
Classification
Sentiment Class
Sentiment Analysis

Introduction

- Sentiment analysis, also referred to as opinion mining, is an approach to natural language processing (NLP) that identifies the emotional tone behind a body of text. This is a popular way for organizations to determine and categorize opinions about a product.

Examples for sendiment analysis

Sentiment analysis is a powerful tool, which can be used across industries and teams. Learn about some of the most popular sentiment analysis business applications, below:

- Social media monitoring
- Brand monitoring
- Customer support analysis
- Customer feedback analysis
- Market research

Data Preprocessing :

Our data generally comes from a variety of different sources and is often in a variety of different formats. For this reason, cleaning our raw data is an essential part of preparing our dataset. However, cleaning is not a simple process, as textual data often contains redundant and/or repetitive words.

Before training the model, we will perform various pre-processing steps on the dataset such as:

- Removing stop words.
- Removing emojis.
- Removing of mentions.
- Removal of numbers.
- Removal of whitespaces.
- Removal of duplicated rows.
- Removal of unuseful columns.
- Converting the text document to lowercase for better generalization.
- Cleaning the ponctuation (to reduce unnecessary noise from the dataset).
- Removing the repeating characters from the words along with removing the URLs/hyperlinks as they do not have any significant importance.

and much more, we will see this in detail later...

We will then performe:

Stemming : reducing the words to their derived stems.

Lemmatization : reducing the derived words to their root form known as lemma for better results.

- **Lowering Case**:

Lowering case is very imprtant since it allows us to make words with same value equal. This will be very useful to reduce the dimensions of our vocabulary.

```
# Lowering Case :
 print("=========== Before Lowering case ============\n")
print("\t" + df.loc[10, "text"])
 print("\n=========== After Lowering case ===========\n")
df['text'] = df['text'].str.lower()
print("\t" + df.loc[10, "text"])
=========== Before Lowering case ============ spring break in
plain city... it's snowing
```

```
========== After Lowering case ===
========
spring break in plain city... it's snowing
```

Lower case was successfully applied to our data

<u>Removal of Mentions:</u>

In social media, Mentions are used to call/mention another user into our post. Generally, mentions don't have an added value to our model. So we will remove them.

A mention has a special pattern: **@UserName**, So we will remove all string which starts with @

- *# Removal of Mentions:*

```
## Creating a fucntion that will be applied to our datset :
 def RemoveMentions(text):
 text_ = re.sub(r"@\S+", "", text)
return text_
## Applying the function to each row of the data
print("========== Before Removing M
entions ==========\n")
print("\t" + df.loc[5, "text"])
print("\n========== After Removing
Mentions ==========\n")
 df["text"] = df["text"].apply(Remove
Mentions)
 print("\t" + df.loc[5, "text"])
```

```
========== Before Lowering case ===
=========
spring break in plain city... it's snowing
 ========== After Lowering case ===
========
spring break in plain city... it's snowing
```

Lower case was successfully applied to our data

Removal of Mentions:

In social media, Mentions are used to call/mention another user into our post. Generally, mentions don't have an added value to our model. So we will remove them.

A mention has a special pattern: **@UserName**, So we will remove all string which starts with @

```
# Removal of Mentions:
 ## Creating a fucntion that will be applied to our datset :
 def RemoveMentions(text):
text_ = re.sub(r"@\S+", "", text)
return text_
## Applying the function to each row of the data print("========== Before Removing M
entions ==========\n")
 print("\t" + df.loc[5, "text"])
print("\n========== After Removing
 Mentions ==========\n")
 df["text"] = df["text"].apply(Remove
Mentions) print("\t" + df.loc[5, "text"])
========== Before Removing Mentions
 ==========
@kwesidei not the whole crew
 ========== After Removing Mentions ==========
not the whole crew
```

Removal of Mentions was successfully applied to our data

Removal of Special Characters:

Special characters are every where, since we have punctuation marks in our tweets. In order to treat, for example, **hello!** and **hello** in the same way. we have to remove the punctuation mark

```
 # Defining a list containing punctuation signs of english :
 punctuations_list = string.punctuation
## Defining that will be applied to our datset :
 def RemovePunctuations(text):
 transformator = str.maketrans
('', '', punctuations_list)
 return text.translate(transformator)
```

```
## Applying the fucntion to all rows : print("===========
Before Removing Punctuations ==============\n")
print("\t" + df.loc[10, "text"]) print("\n===========
After Removing
 Punctuations \===========\n")
 df["text"] = df["text"].apply(RemovePunctuations)
print("\t" + df.loc[10."text"])
========== Before Removing Punctuations ==============
spring break in plain city...
it's snowing
 ========== After Removing Punctuations \===========
 spring break in plain city its snowing
```

- **Removal of Stop words**:
- Stopwords are the most common words in any natural language. For the purpose of analyzing text data and building NLP models, these stopwords might not add much value to the meaning of the document.
- Generally, the most common words used in a text are “the”, “is”, “in”, “for”, “where”, “when”, “to”, “at” etc.
- Consider this text string – “There is a pen on the table”. Now, the words “is”, “a”, “on”, and “the” add no meaning to the statement while parsing it. Whereas words like “there”, “book”, and “table” are the keywords and tell us what the statement is all about.

Loading the Dataset :

In order to build our classifier model, we need a dataset which contains a huge number of tweets and the corresponding feeling being expressed at.

In any project related to the manipulation and analysis of data, we always start by collecting the data on which we are going to work. In our case, we will import our data from a .csv file. The dataset provided is the Sentiment140 Dataset which consists of 1,600,000 tweets that have been extracted using the Twitter API.

The various columns present in the dataset are:

target: the polarity of the tweet (positive or negative)

ids: Unique id of the tweet

date: the date of the tweet

flag: It refers to the query. If no such query exists then it is NO QUERY.

user: It refers to the name of the user that tweeted

text: It refers to the text of the tweet

Input:

```
# Importing the dataset :
DATASET_COLUMNS=['target','ids','date','flag','user','text']
DATASET_ENCODING = "ISO-8859-1“

Df=pd.read_csv('../input/tweets/training.1600000.processed.noemoticon.csv', encoding=DATASET_ENCODING, names=DATASET_COLUMNS)

# Display of the first 5 lines : df.sample(5)
```

Output

	Target	Ids	Date	Flag	User	Text
553277	0	2203589156	Wed Jun 17 00:04:43 PDT 2009	NO_QUERY	Ausste	Art Center assignment due in two weeks, Im fee…
1077806	4	1967737627	Fri May 29 20:07:22 PDT 2009	NO_QUERY	trishysabel	theres crickets in the pepsi center. u guys …
784549	0	2324197774	Thu Jun 25 02:32:28 PDT 2009	NO_QUERY	aamy23	wishing i was at the coast, sitting by the poo…
1157521	4	1979188875	Sun May 31 01:59:52 PDT 2009	NO_QUERY	poposkidimitar	Leaving Zadar, heading to Zagreb this afternoon
197871	0	1971093961	Sat May 30 06:11:43 PDT 2009	NO_QUERY	MYCBA	in london at last. missed the connecting fligh…

Conclusion

Sentiment analysis is a technique used to understand the emotional tone of the text. It can be used to identify positive, negative, and neutral sentiments in a piece of writing.

Thank you