

Quick Guide: Steps To Perform Text Data Cleaning in Python

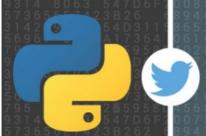
BEGINNER BUSINESS ANALYTICS INFOGRAPHIC INFOGRAPHICS NLP PYTHON TECHNIQUE TEXT UNSTRUCTURED DATA

Introduction

Twitter has become an inevitable channel for brand management. It has compelled brands to become more responsive to their customers. On the other hand, the damage it would cause can't be undone. The 140 character tweets has now become a powerful tool for customers / users to directly convey messages to brands.

For companies, these tweets carry a lot of information like sentiment, engagement, reviews and features of its products and what not. However, mining these tweets isn't easy. Why? Because, before you mine this data, you need to perform a lot of cleaning. These tweets, once extracted can come with unwanted html characters, bad grammar and poor spellings – making the mining very difficult.

Below is the infographic, which displays the steps of cleaning this data related to tweets before mining them. While the example in use is of Twitter, you can of course apply these methods to any text mining problem. We've used Python to execute these cleaning steps.



Effective Text Data Cleaning using Python

Benefits of mining for a brand?

You can do sentimental analysis to discover customer's sentiment for a brand

You can measure brand popularity using the actively engaged tweeters

It is used to identify the pain points of customers i.e. customer relationship management

It is widely used for predictions and forecasting



The Business Problem

Let's say, we want to find the features of an Apple iPhone which are most popular amongst the fans on Twitter.

What to do next?

We've extracted all the tweets related to consumer opinions of iPhone. Here's a sample tweet on which we'll perform data cleaning

TWEET

"I luv my 6lt;3 iphone 6amp; you're awsm apple. DisplayIsAwesome, sooo happppppy:) http://www.apple.com"

Steps for Data Cleaning

STEP 01

Escaping HTML characters

Code

import HTMLParser html parser = HTMLParser.HTMLParser() tweet = html_parser.unescape(original_tweet)



Output

» "I luv my <3 iphone & you're awsm apple. Display Is Awesome, sooo happppppy http://www.apple.com"

Decoding data

STEP

tweet = original_tweet.decode("utf8").encode('ascii','ignore')

Output

"I luv my «3 iphone & you're awsm apple. DisplayIsAwesome, sooo happppppy:) http://www.apple.com"

STEP 03

Apostrophe Lookup

Code

APPOSTOPHES = {"'s": " is", "'re": " are", ...} ## Need a huge dictionary words = tweet.split()

reformed = [APPOSTOPHES[word] if word in APPOSTOPHES else word for word in words]
reformed = " ".join(reformed)

Outcome

"I luv my <3 iphone & you are awsm apple. DisplayIsAwesome, sooo happppppy:) http://www.apple.com"

Removal of Stop-Words

STEP 04

When data analysis needs to be data driven at the word level, the commonly occurring words (stop-words) should be removed.

One can either create a long list of stop-words or one can use predefined language specific libraries.

5TEP 05

Removal of Punctuations

All the punctuation marks according to the priorities should be dealt with. For example: ".", ",","?" are important punctuations that should be retained while others need to be removed.

Removal of Expressions

STEP **06**

Textual data (usually speech transcripts) may contain human expressions like [laughing], [Crying], [Audience paused]. These expressions are usually non relevant to content of the speech and hence need to be removed.

STEP **07**

Split Attached Words

Code

cleaned = " ".join(re.findall('[A-Z][^A-Z]*', original_tweet))

Outcome

"I luv my <3 iphone & you are awsm apple. Display Is Awesome, sooo happppppy:) http://www.apple.com"</p>

Slangs lookup

STEP 08

Code

tweet = slang loopup(tweet)

Outcome

"I love my <3 iphone & you are awesome apple. Display Is Awesome, sooo happppppy:) http://www.apple.com"

> STEP 09

Standardizing word

Code

tweet = ".join(".join(s)[:2] for _, s in itertools.groupby(tweet))

Outcome

"I love my <3 iphone & you are awesome apple. Display Is Awesome, so happy:) http://www.apple.com"

Removal of URLs

STEP

URLs and hyperlinks in text data like comments, reviews, and tweets should be removed.

Final cleaned tweet:

>> "I love my iphone & you are awesome apple. Display Is Awesome, so

Advanced Data Cleaning

Grammar checking

Grammar checking is majorly learning based, huge amount of proper text data is learned and models are created. Many online tools are available for grammar correction purposes.

Spelling correction

In natural language, misspelled errors are encountered. One can use algorithms like the Levenshtein Distances, Dictionary Lookup etc. other modules and packages to fix these errors.

Your Next Steps...

Now that the data (tweet) is cleaned, you are ready to practice and learn the following techniques (in no order) of Text Mining-

- 1. Framework to build a niche dictionary for text mining http://bit.lu/leetMw6
- 2 Step by Step guide to extract insights from free text http://bit.ly/1JjslYe
- 3. 2014 FIFA World Cup Prediction using Twitter Mining http://bit.ly/1kLeYSk
- 4. Text Mining Hack using Google API http://bit.ly/1LDPF6c



For more resources on analytics/data science, visit

www.analyticsvidhya.com



Download the PDF Version of this infographic and refer the python codes to perform Text Mining and follow your 'Next Steps...' -> <u>Download Here</u>

If you like what you just read & want to continue your analytics learning, subscribe to our emails, follow us on twitter or like our facebook page.

Article Url - https://www.analyticsvidhya.com/blog/2015/06/quick-guide-text-data-cleaning-python/



Analytics Vidhya

This is the official account of the Analytics Vidhya team.