Challenges:

First Challenge was on what basis should the data be tokenized

How to handle punctuations like “.,!?/” etc

How to recognize mentions, hashtags, urls, email

Handling emoticons which are appearing as unicode like and its hard to even recognize and classify

How to handle incomplete urls

How to handle url like tokens ex: https... where the last 3 dots are displaying as unicode and are wrong being classified as urls.

Given that data is very random without any grammatical structure, with frequent urls, its impossible to handle all variants of data or even recognize all variants given the huge size of data. Handling them would demand manual check of all special cases.

Design:

Used space (“ ”) as token seperator.

For a couple of reasons

1. In general words that occur together specify a different meaning compared to when they are seperated. For ex: hell!! and hell in following sentences “what the hell!!” and “Way to hell is clear” are different and carry a semantic meaning with it, which will be lost if “hell!!” is seperated on punctuations and becones “hell”

2. Also, as discussed in class “ram.” is different token and “ram” is a different token.

3. So, tokenizing on anything except space will modify and may result in loss of info.

Not seperating “:” from “@someName:”

1. Reason is simple enough, while jst a mention (@tag) refers to a person, mention+”:” , specifies he/she is the owner of a tweet which was retweeted.

2. So, if split based on “:”, queries like how many retweets of a particular person have been done cant be answered, implies data is lost.

Since removing noise in not part of tokenization, most frequent words like RT, are allowed are not removed.

Hashtags are recognized by comparing first character of a token to “#”

Mentions are recognized by comparing first character of a token to “@”

Urls are recognized by comparing first 4 characters to “http”
Emails are recognized by comparing last 3 charcaters of a token to “com”
Since token is first checked for urls and then for emails, urls with “com” ending will not be mistakenly classified as emails.