Sentiment analysis using TextAnalysis.jl

This notebook explores the **Twitter US Airline Sentiment** data set from Kaggle. We will use **TextAnalysis.jl** as the primary tool for analyzing textual data.

```
using Pkg
```

dir = "/Users/tomkwong/Julia/HumansOfJulia-WeeklyContest/Week2-TextAnalysis.jl/tk3369"

```
dir = "/Users/tomkwong/Julia/HumansOfJulia-WeeklyContest/Week2-TextAnalysis.jl/tk3369"
```

```
Pkg.activate(dir)
```

```
begin
    using TextAnalysis
    using CSV
    using DataFrames
    using Pipe: @pipe
    using Plots
end
```

Loading data

```
cd(dir)
```

df =

	tweet_id	airline_sentiment	airline_sentiment_confidence	negativereason
1	570306133677760513	"neutral"	1.0	missing
2	570301130888122368	"positive"	0.3486	missing
3	570301083672813571	"neutral"	0.6837	missing
4	570301031407624196	"negative"	1.0	"Bad Flight"
5	570300817074462722	"negative"	1.0	"Can't Tell"
6	570300767074181121	"negative"	1.0	"Can't Tell"
7	570300616901320704	"positive"	0.6745	missing

	tweet_id	airline_sentiment	airline_sentiment_confidence	negativereason
8	570300248553349120	"neutral"	0.634	missing
9	570299953286942721	"positive"	0.6559	missing
10	570295459631263746	"positive"	1.0	missing
more				
14640	569587140490866689	"neutral"	0.6771	missing

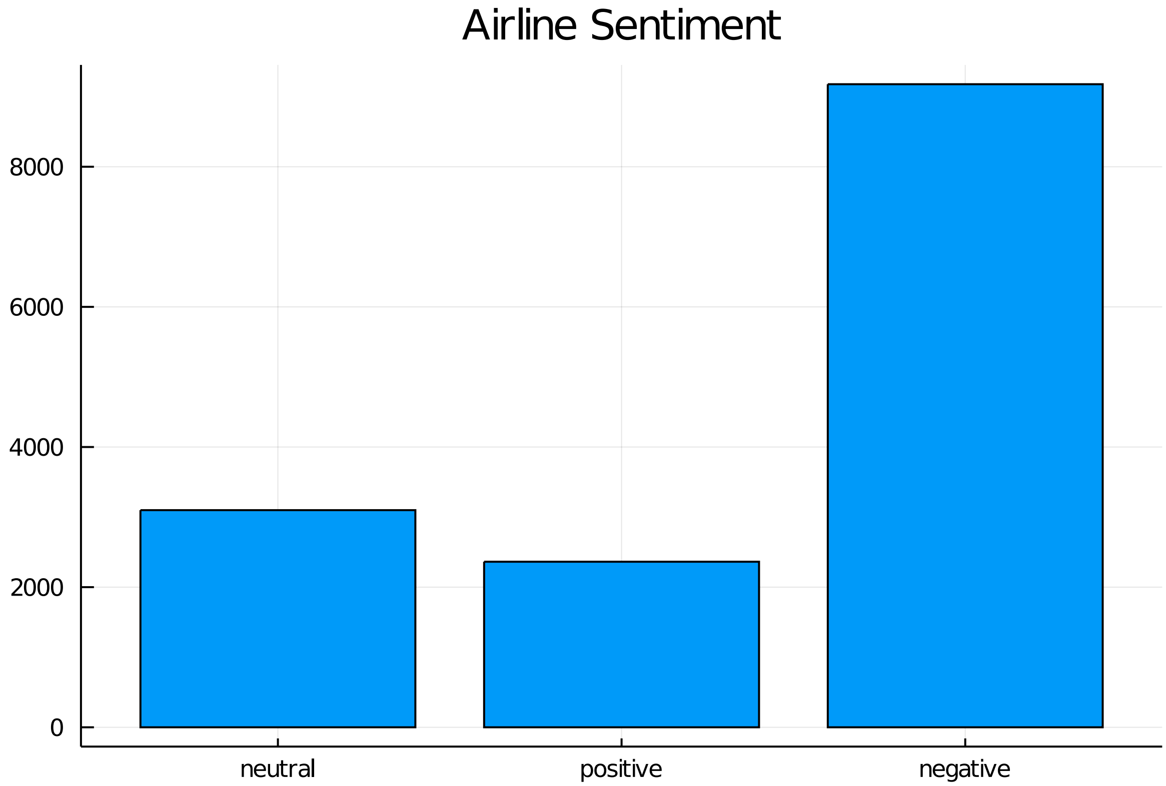
```
df = DataFrame(CSV.File("data/Tweets.csv"))
```

Data Wrangling

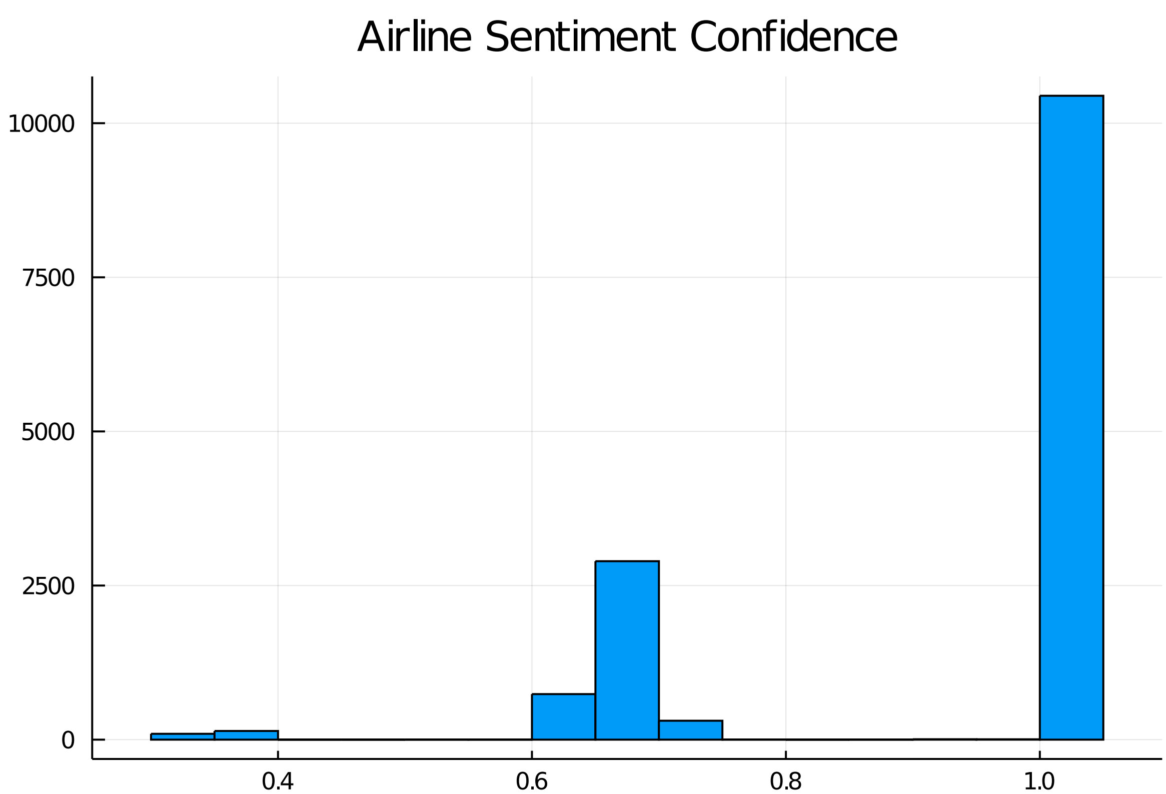
We will take a look at the data and get a little more understanding about what's going on in this data set.

	variable	eltype	nmissing	fi
1	:tweet_id	Int64	nothing	570306133677760513
2	:airline_sentiment	String	nothing	"neutral"
3	:airline_sentiment_confidence	Float64	nothing	1.0
4	:negativereason	Union{Missing, String}	5462	"Bad Flight"
5	:negativereason_confidence	Union{Missing, Float64}	4118	0.0
6	:airline	String	nothing	"Virgin America"
7	:airline_sentiment_gold	Union{Missing, String}	14600	"negative"
8	:name	String	nothing	"cairdin"
9	:negativereason_gold	Union{Missing, String}	14608	"Late Flight\nFlig aints"
10	:retweet_count	Int64	nothing	0
11	:text	String	nothing	"@VirginAmerica Wh d."
12	:tweet_coord	Union{Missing, String}	13621	"[40.74804263, -73
13	:tweet_created	String	nothing	"2015-02-24 11:35:
14	:tweet_location	Union{Missing, String}	4733	"Lets Play"
15	:user_timezone	Union{Missing, String}	4820	"Eastern Time (US

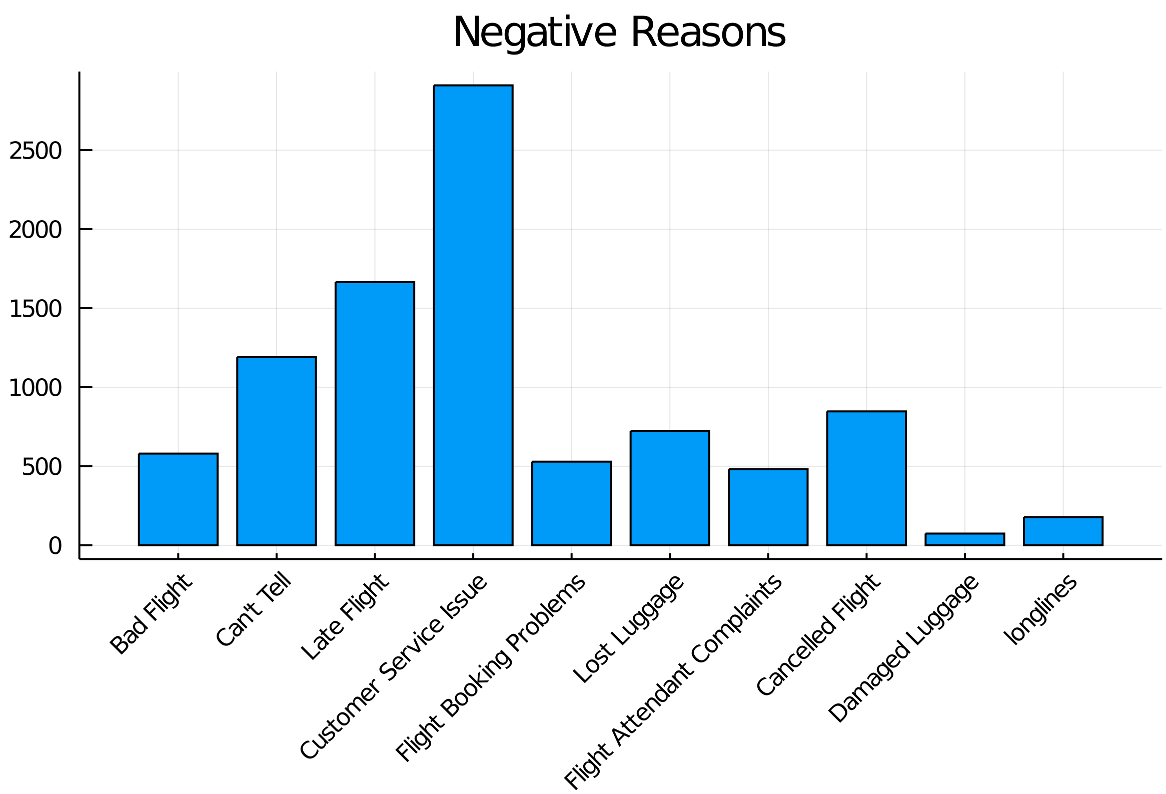
```
describe(df, :eltype, :nmissing, :first => first)
```




```
let x = combine(groupby(df, :airline_sentiment), nrow)
    bar(x.airline_sentiment, x.nrow;
        title = "Airline Sentiment",
        label = :none,
        legend = :topright)
end
```




```
histogram(df.airline_sentiment_confidence;
    legend = nothing,
    title = "Airline Sentiment Confidence")
```




```
let x = combine(groupby(dropmissing(df, :negativereason), :negativereason), nrow)
    bar(x.negativereason, x.nrow;
        title = "Negative Reasons", label = :none, xrotation = 45)
end
```

Examining tweets

The CSV file contains over 14,000 tweets. Let's quickly examine some individual data.

Before we go further, it would be nice to display a single record in table format. We can define a `table` function that converts an indexable object into Markdown format, which can be displayed in this Pluto notebook.

```
function table(nt)
    io = IOBuffer()
    println(io, "|name|value|")
    println(io, "|---:|:----|")
    for k in keys(nt)
        println(io, "|`", k, "`|", nt[k], "|")
    end
    return Markdown.parse(String(take!(io)))
end;
```

Here, we will define a variable called `row` and bind it to a slider for quick experimentation.

```
@bind row html"""<input type="range" min="1" max="100" value="36"/>"""
```

```
"Current Record: 36"
```

```
"Current Record: $row"
```

name	value
tweet_id	570051991277342720
airline_sentiment	neutral
airline_sentiment_confidence	0.6207
negativereason	missing
negativereason_confidence	missing
airline	Virgin America
airline_sentiment_gold	missing
name	miaerolinea
negativereason_gold	missing
retweet_count	0
text	Nice RT @VirginAmerica: Vibe with the moodlight from takeoff to touchdown. #MoodlitMonday #ScienceBehindTheExperience http://t.co/Y7O0uNxTQP
tweet_coord	missing
tweet_created	2015-02-23 18:46:00 -0800
tweet_location	Worldwide
user_timezone	Caracas

```
table(df[row, :])
```

As an example, record #36 has the tweet text as:

```
Nice RT @VirginAmerica: Vibe with the moodlight from takeoff to touchdown. #MoodlitM
onday #ScienceBehindTheExperience http://t.co/Y7O0uNxTQP
```

This is a tricky one because it contains all of the followings:

- mention (`@VirginAmeria`)
- hash tag (`#MoodlitMonday` and `#ScienceBehindTheExperience`)
- URL (`http://t.co/Y7O0uNxTQP`)

Technically `RT` is a shorthand for "retweet" so perhaps it should be expanded but let's not worry about that for now.

Handling mentions and hashtags

If we just ignore these problems then it can be a disaster.

name	value
moodlight	1
unxtqp	1
vibe	1
moodlitmonday	1
virginamerica	1
takeoff	1
sciencebehindtheexperi	1
nice	1
httpcoy	1
o	1
rt	1
7	1
touchdown	1
0	1

```
let s = df[36, :text]
    sd = StringDocument(lowercase(s))
    op = 0x00
    op |= strip_punctuation
    op |= strip_stopwords
    op |= strip_html_tags
    prepare!(sd, op)
    stem!(sd)
    table(ngrams(sd))
end
```

Right off the bat, I can see some problems here. It seems that when I stripped punctuations, it also took the `@` and `#` signs away. The URL also became weird. Oh yeah, that's what stripping punctuation means, right? :-)

Extracting mentions, hash tags, and URL's.

This neat idea came from José Bayoán Santiago Calderón when I asked the question on Slack. Let's define some functions using regular expressions.

```
const regexp = Dict(
    :mention => r"@\w+",
    :hashtag => r"#\w+",
    :url => r"http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[!*\(\),]|(?:%[0-9a-fA-F][0-9a-
fA-F]))+"
);
```

```
extract_tokens (generic function with 1 method)
```

```
function extract_tokens(s, token_type)
    return collect(x.match for x in eachmatch(regexp[token_type], s))
end
```

```
remove_tokens (generic function with 1 method)
```

```
function remove_tokens(s)
    for re in values(regexp)
```

```
        s = replace(s, re => "")
    end
    return s
end
```

Create a new data frame with extracted and clean text fields

```
begin
    df2 = DataFrame()
    df2.airline_sentiment = df.airline_sentiment
    df2.text = df.text
    df2.mentions = extract_tokens.(df.text, :mention)
    df2.hashtags = extract_tokens.(df.text, :hashtag)
    df2.urls = extract_tokens.(df.text, :url)
    df2.clean_text = lowercase.(remove_tokens.(df.text))
    df2
end;
```

name	value
airline_sentiment	neutral
text	Nice RT @VirginAmerica: Vibe with the moodlight from takeoff to touchdown. #MoodlitMonday #ScienceBehindTheExperience http://t.co/Y7O0uNxTQP
mentions	SubString{String}["@VirginAmerica"]
hashtags	SubString{String}["#MoodlitMonday", "#ScienceBehindTheExperience"]
urls	SubString{String}["http://t.co/Y7O0uNxTQP"]
clean_text	nice rt : vibe with the moodlight from takeoff to touchdown.

```
table(df2[36, :])
```

As you can see, the mentions/hashtags/urls are extracted into separate columns in the data frame. The `clean_text` field contains the cleaned version of `text`.

Using Naive Bayes Classifier

In our data frame, we already have a column `x_string_doc` with `StringDocuments` values. So we can just fit them to the classifier.

```
using TextAnalysis: NaiveBayesClassifier, fit!, predict
```

create_string_doc (generic function with 1 method)

```
function create_string_doc(s)
    sd = StringDocument(s)
    op = 0x00
    op |= strip_punctuation
    op |= strip_stopwords
    op |= strip_html_tags
    prepare!(sd, op)
    stem!(sd)
```

```
    return sd
end
```

```
model = let
    classes = unique(df2.airline_sentiment)
    nbc = NaiveBayesClassifier(classes)
    for (clean_text, class) in zip(df2.clean_text , df2.airline_sentiment)
        sd = create_string_doc(clean_text)
        fit!(nbc, sd, class)
    end
    nbc
end;
```

Let's create a model test function and then try our predictor for a few simple test cases.

```
function test_model(model, tweets)
    df = DataFrame(text = tweets)
    df.doc = TextAnalysis.text.(create_string_doc.(remove_tokens.(tweets)))
    df.analysis = predict.(Ref(model), df.doc)

    df.positive = getindex.(df.analysis, "positive")
    df.negative = getindex.(df.analysis, "negative")
    df.neutral = getindex.(df.analysis, "neutral")

    select!(df, Not(:analysis))

    return df
end;
```

	text	doc	positive	negative	neutral
1	"whatever airline sucks!"	"whatev airlin suck"	0.112082	0.852377	0.0355411
2	"i love @american service :-)"	"love servic"	0.795714	0.136063	0.0682232
3	"just ok"	"ok"	0.462275	0.201953	0.335771
4	"hello world"	"hello world"	0.186748	0.120565	0.692687

```
let
    tweets = [
        "whatever airline sucks!",
        "i love @american service :-)",
        "just ok",
        "hello world"]
    test_model(model, tweets)
end
```

Determining accuracy

How well does the Naive Bayes Classifier work?

As the `predict` function returns a `Dict` object with the probabilities assigned to each class, we need to choose the best option. Let's define a function for that.

```
function predict_and_choose(c::NaiveBayesClassifier, sd::StringDocument)
    val = predict(c, sd)
    return argmax(val)
end;
```

Now, make prediction over all 14K tweets.

```
yhat = let sds = create_string_doc.(lowercase.(remove_tokens.(df2.text)))
    predict_and_choose.(Ref(model), sds)
end;
```

hits = 12104

```
hits = count(df2.airline_sentiment .== yhat)
```

misses = 2536

```
misses = length(yhat) - hits
```

wayoff = 733

```
wayoff = count(
            (df2.airline_sentiment .!== yhat) .&
            (df2.airline_sentiment .!== "neutral") .&
            (yhat .!== "neutral"))
```

accuracy_percentage = 82.6775956284153

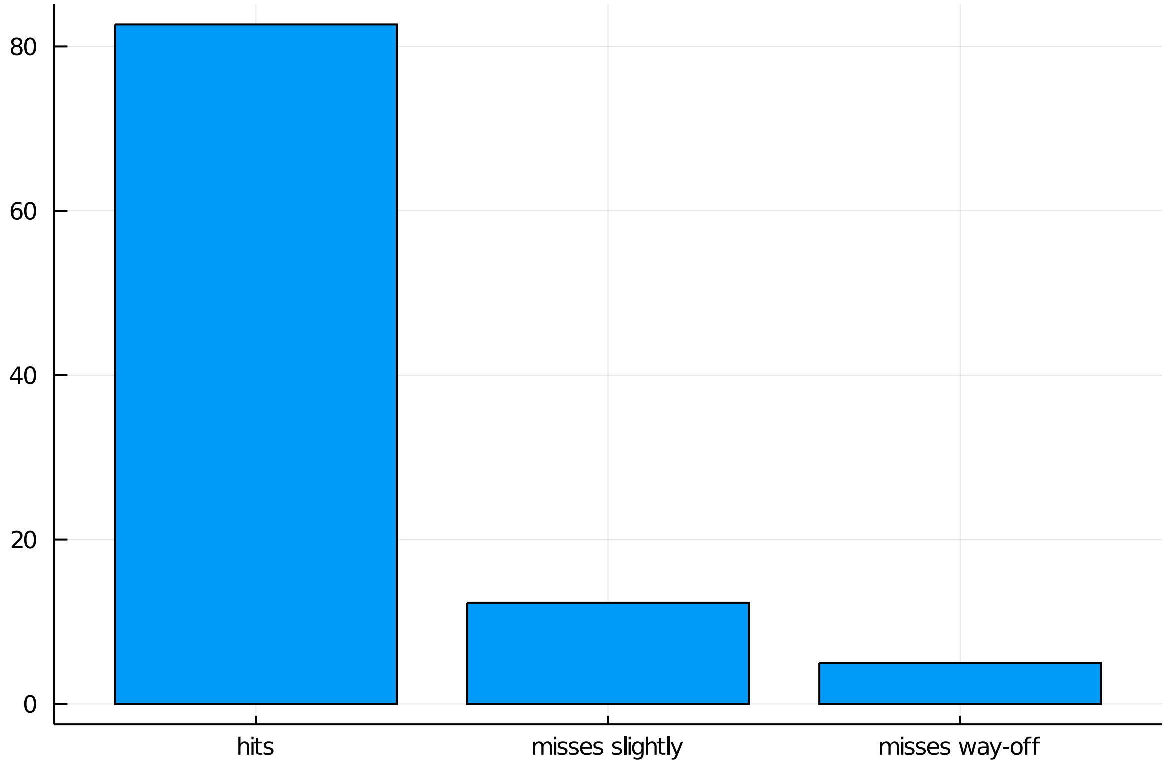
```
accuracy_percentage = hits / (hits + misses) * 100
```

slightly_off_percentage = 12.315573770491802

```
slightly_off_percentage = (misses - wayoff) / (hits + misses) * 100
```

way_off_percentage = 5.006830601092896

```
way_off_percentage = wayoff / (hits + misses) * 100
```




```
bar(["hits", "misses slightly", "misses way-off"],
    [accuracy_percentage, slightly_off_percentage , way_off_percentage];
    legend = :none)
```

Analyze some random tweets

```
begin
    using HTTP
    using JSON3
end
```

```
token = readline("/Users/tomkwong/.twitter-bearer");
```

```
response = HTTP.get("https://api.twitter.com/1.1/search/tweets.json?
q=lang%3Aen%20flight", ["authorization" => "Bearer $token"]);
```

```
data = JSON3.read(response.body);
```

```
Object(:created_at ⇒ "Thu Nov 12 09:28:47 +0000 2020", :id ⇒ 1326819219015667712, :id
```

```
data[:statuses][1]
```

```
tweets =
  String["RT @TheClub_Lounge: Don't forget to fuel up before your flight! Guests at The Clu
```

```
tweets = [x.text for x in data.statuses]
```

	text	doc	positi
5	"@nbamaryyy Ratio for the legend Flight https://t.co/xxcyspBbSy"	"Ratio legend Flight"	0.21522
6	"RT @LMKMovieManiac: Uriyadi @Vijay_B_Kumar dialogs, Saattai Adi👌 \n\nMaara to Balayya (Mallya) - \"You are a socialite, I'm a socialist\"\n\n\"Yen…"	"RT Uriyadi dialog Saattai Adi👌 Maara Balayya Mallya You socialit I socialist Yen …"	0.16625
7	"Airbus is developing a system, called fello'fly, in which two commercial planes mimic migrating birds by travelling… https://t.co/Q4e2VrLxdv"	"Airbus develop system call fellofli commerci plane mimic migrat bird travel …"	0.05694
8	"@F1 Remove the one make W series from the race weekend & you won't have to transport them around the world. Then yo… https://t.co/tNo7Y886VX"	"Remov W seri race weekend amp won transport world Then yo …"	0.44049
9	"@AuroraEstella A sort of motorway service station stop off before they reach Heathrow to check in for the flight. Makes sense."	"A sort motorway servic station stop reach Heathrow check flight Make sens"	0.01374
10	"RT @kg_suresh: Unprofessional @IndiGo6E. Booked veg meal on Bho-Del flight. Air hostess says no booking. When shown ticket, she says airlin…"	"RT Unprofession Book veg meal BhoDel flight Air hostess book When shown ticket airlin …"	0.00304
11	"@lcveslix flight attendant"	"flight attend"	0.34917
12	"RT @JetTipNet: Bahrain Royal Flight 747-400, A9C-HAK, apparently here at MSP to repatriate the remains of Prime Minister Prince Khalifa bin…"	"RT Bahrain Royal Flight 747400 A 9 CHAK appar MSP repatri remain Prime Minist Princ Khalifa bin …"	0.00119
13	"SAGITTARIUS \n⋆ can you just take a break?\n⋆ self-care is booking a last-minute flight at 2am\n⋆ I'm shocked you're s… https://t.co/qmo64UD0Y9"	"SAGITTARIUS ⋆ break ⋆ selfcar book minut flight 2 am ⋆ I shock re …"	0.00179
14	"RT @aadhavkk: That amazing feel aftr u watch a good film! All i can do is thank the team of #SooraraiPottru for giving us this gem 👏 @Suri…"	"RT That amaz feel aftr watch film All thank team give gem 👏 …"	0.99030
15	"RT @MaryamNSharif: On board the flight to Islamabad. Thank you GB for your unprecedented love, affection and support. Will repay with more…"	"RT On board flight Islamabad Thank GB unpreced love affect support Will repay …"	0.54975

```
result = test_model(model, tweets)
```

name	value
text	@AuroraEstella A sort of motorway service station stop off before they reach Heathrow to check in for the flight. Makes sense.
doc	A sort motorway servic station stop reach Heathrow check flight Make sens
positive	0.013747470658715571
negative	0.9778303914428919
neutral	0.008422137898392615

```
table(result[9,:])
```