Sentiment Analysis

The procedure of computationally distinguishing and classifying assessments expresses as texts, particularly to decide if the writer's attitude towards a specific theme, item, situation etc. is positive, negative, or impartial

Types of Sentiment Analysis:

Sentiment analysis models focus on polarity (positive, negative, neutral) but also on feelings and emotions (angry, happy, sad, etc.), and even on intentions (e.g. *interested* v. *not interested*).

Fined gained:

If polarity precision is important to your business, you might consider expanding your polarity categories to include:

😊Very positive

😊Positive

😐Neutral

☹Negative

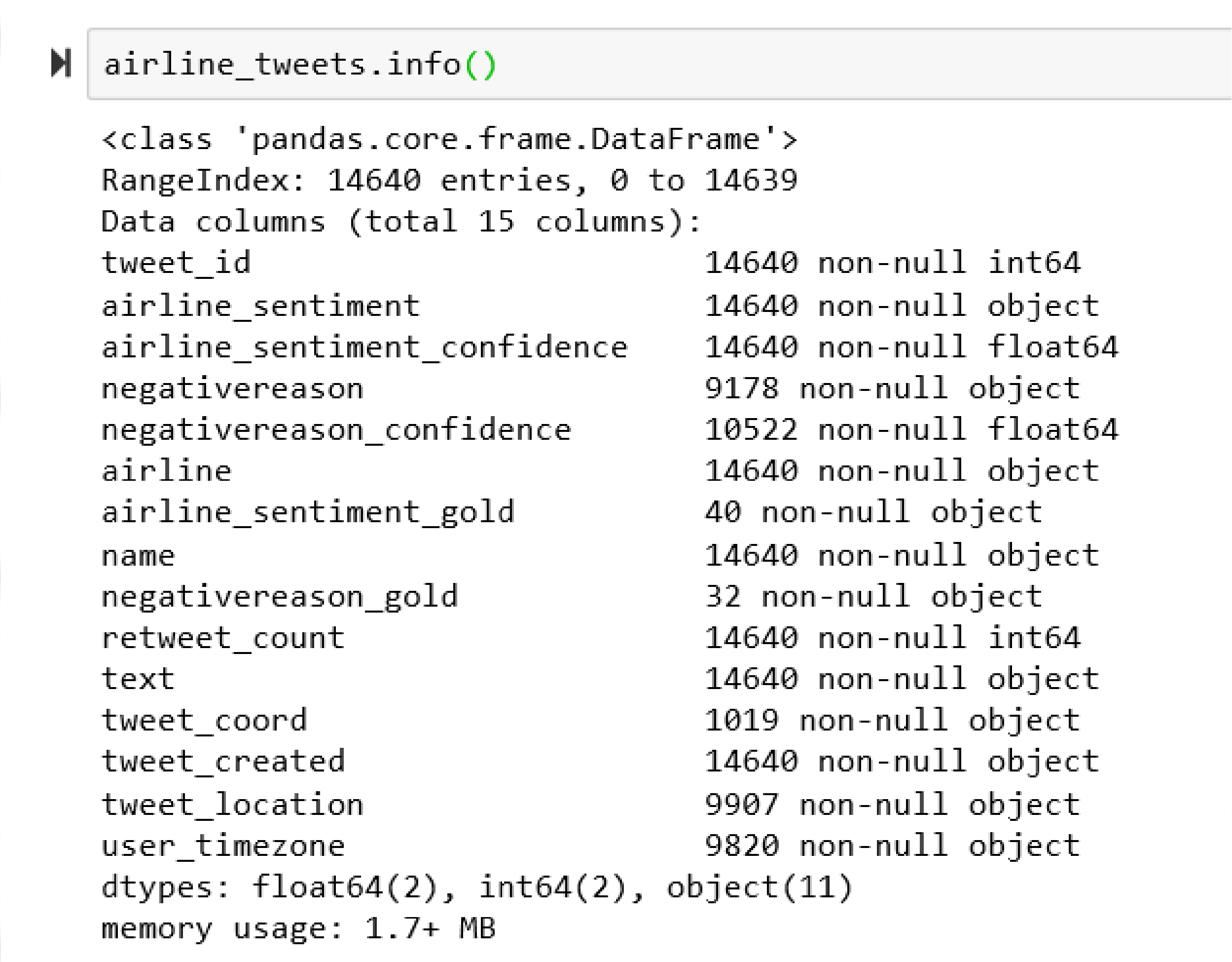
😟Very negative

- **Emotion detection:**
- Specifically focuses on detecting emotions like happiness, frustration, anger, sadness, and so on. Many emotion detection systems use lexicons (i.e. lists of words and the emotions they convey) or complex machine learning algorithms.
- **Aspect-based Sentiment Analysis:**
- Specially used to analyze reviews pertaining to a feature. Ex- The camera is of advanced quality in iPhone.
- **Multilingual sentiment analysis:**
- Detecting text automatically with language classifier, then train a customer sentiment analysis model to classify texts in the language of your choice.

Dataset Description:

For our project we have taken the following datasets:

Airline reviews:



```
airline_tweets.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14640 entries, 0 to 14639
Data columns (total 15 columns):
tweet_id                        14640 non-null int64
airline_sentiment               14640 non-null object
airline_sentiment_confidence    14640 non-null float64
negativereason                  9178 non-null object
negativereason_confidence       10522 non-null float64
airline                         14640 non-null object
airline_sentiment_gold          40 non-null object
name                            14640 non-null object
negativereason_gold             32 non-null object
retweet_count                   14640 non-null int64
text                            14640 non-null object
tweet_coord                     1019 non-null object
tweet_created                   14640 non-null object
tweet_location                  9907 non-null object
user_timezone                   9820 non-null object
dtypes: float64(2), int64(2), object(11)
memory usage: 1.7+ MB
```

Twitter: It is text file containing tweets of million of customers.

Movie reviews: It consists of 2000 words, 1000 positive and 1000 negative

Methods to perform sentiment analysis:

There are many ways to perform sentimental analysis, but the following three are main.

Rule base:

Rule base system uses a set of human crafted rules to help identify subjectivity, polarize, or the subject of an opinion. These rules may include various techniques developed in computational linguistics,

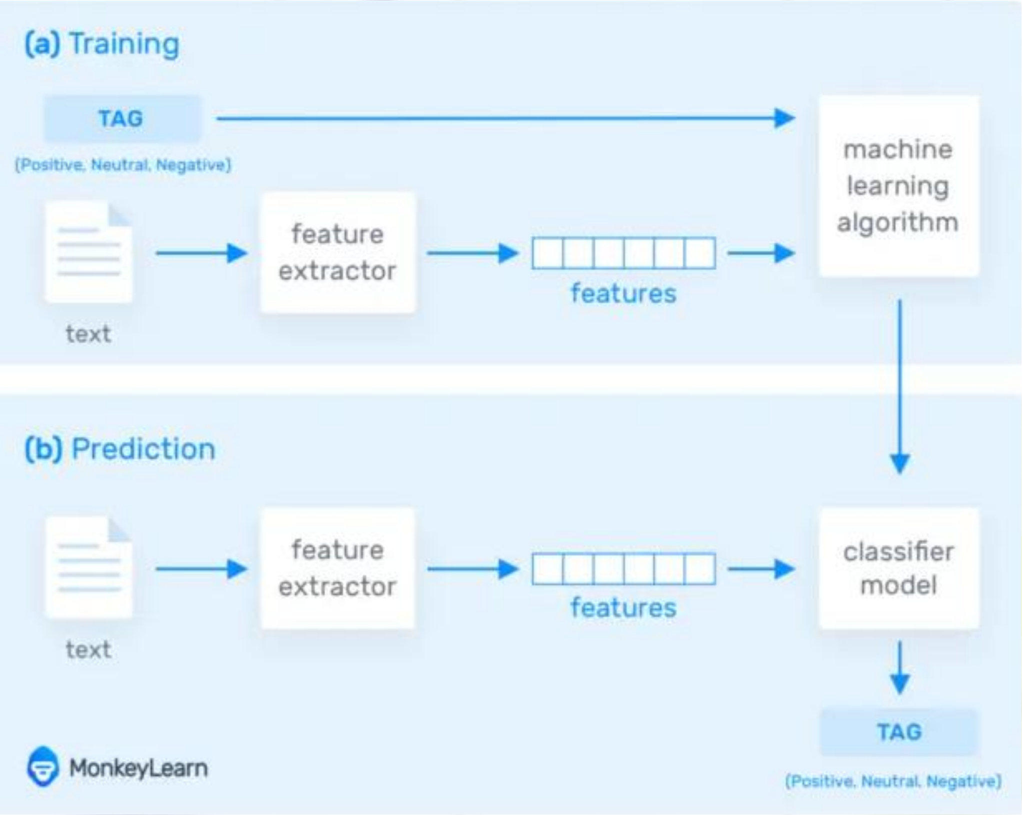
- Stemming
- Tokenization

Let suppose we have two group of words, one with the positive and the other with the negative words. Rule base will look at the count of both types of words, if both are equal in counts then it will say neutral, in case more positive then result will be positive and vice versa. However, in general these are not considered as accurate as many Automatics are

Automatic

This method is used in machine learning when performing sentiment analysis. It is usually modeled as a classification problem, where a classifier is given a text and returns a category.

Eg: Positive, negative



Hybrid:

Hybrid systems that combine both rule-based and automatic approaches

Methodology for our project:

First, we defined a criterion which is known as “feature” so that the classifier distinguishes text and categorized them. For cleaning the data i.e. removing punctuation and stop words we used “Stopwords”. After looping the features, a separate training set and validation set was created (20%).

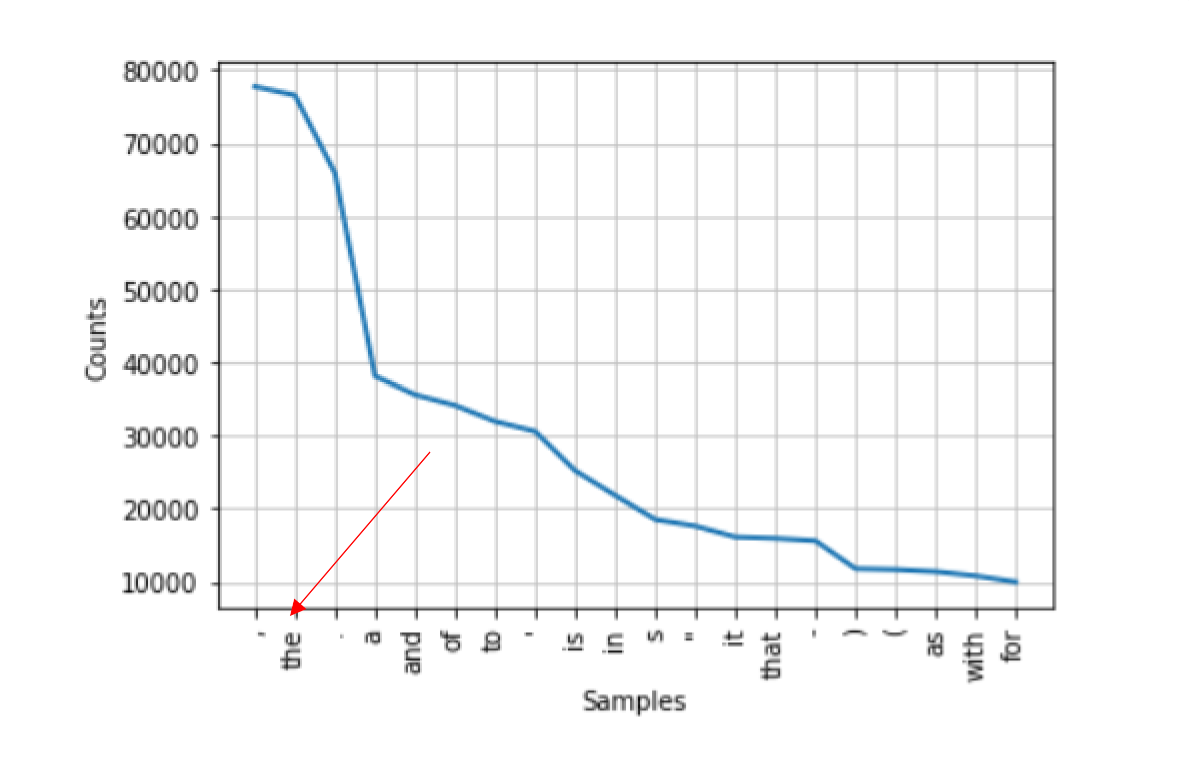
Bag of words feature: All useful words of each review are used to create a feature set. Fixed number of positive and negative reviews for test and train, which results in equal distribution. It extracts only unigram (n-gram of size 1) from the feature set.

Bag of Ngram feature: It extracts only bigram (n-gram of size 2) for the feature set.

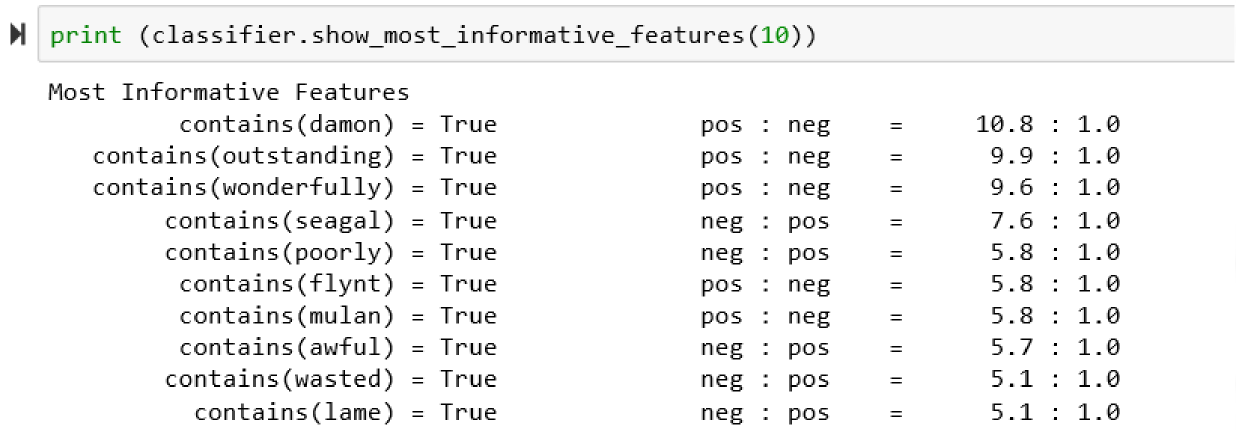
The classifier was trained using different classifiers namely Naive Bayes Classifier, Maximum Entropy Classifier, Decision Tree Classifier, Support Vector Machine Classifier, etc. The accuracy value changes each time you run the program because of the names array being shuffled above.

Result & Observation of Movie reviews:

1) **Most common words:**



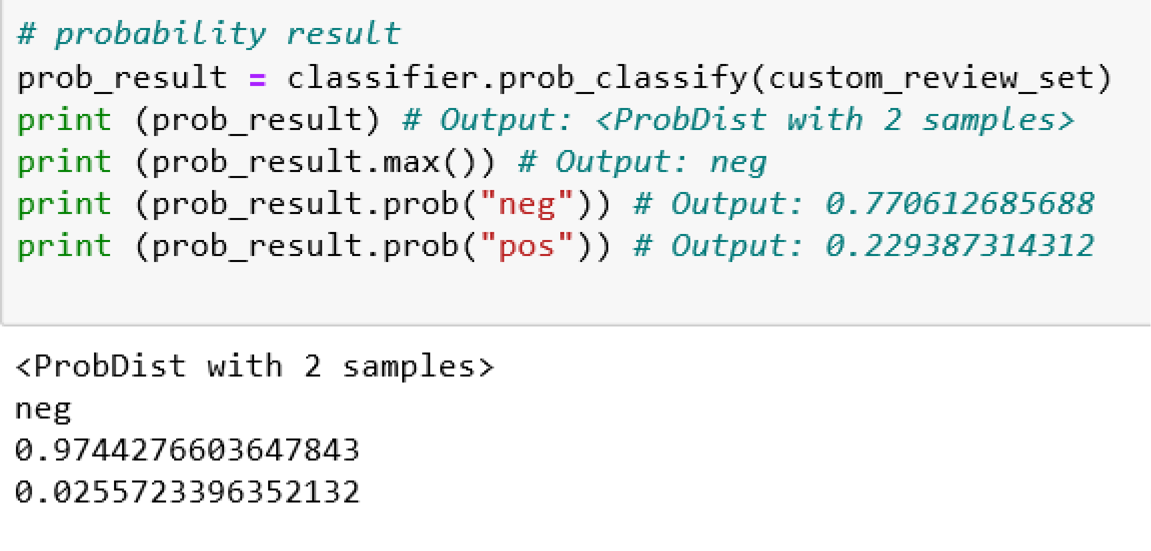
2) **The result shows that the word contain is used 10.8 times more often in positive reviews than in negative reviews.**



```
print (classifier.show_most_informative_features(10))
```

```
Most Informative Features
           contains(damon) = True              pos : neg    =     10.8 : 1.0
     contains(outstanding) = True              pos : neg    =      9.9 : 1.0
     contains(wonderfully) = True              pos : neg    =      9.6 : 1.0
          contains(seagal) = True              neg : pos    =      7.6 : 1.0
          contains(poorly) = True              neg : pos    =      5.8 : 1.0
           contains(flynt) = True              pos : neg    =      5.8 : 1.0
           contains(mulan) = True              pos : neg    =      5.8 : 1.0
           contains(awful) = True              neg : pos    =      5.7 : 1.0
          contains(wasted) = True              neg : pos    =      5.1 : 1.0
            contains(lame) = True              neg : pos    =      5.1 : 1.0
```

3) Probability of negative and positive words:

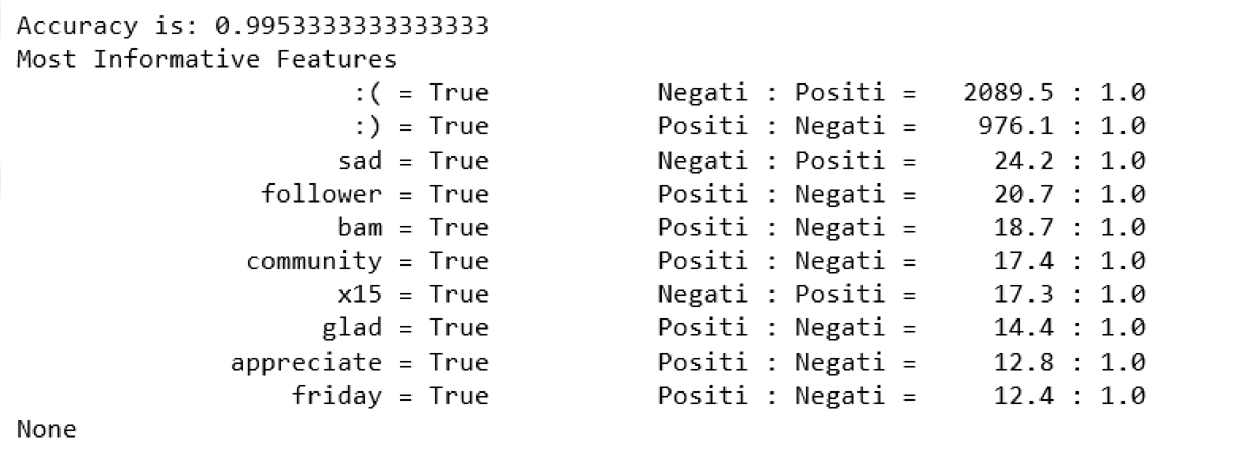


```
# probability result
prob_result = classifier.prob_classify(custom_review_set)
print (prob_result) # Output: <ProbDist with 2 samples>
print (prob_result.max()) # Output: neg
print (prob_result.prob("neg")) # Output: 0.770612685688
print (prob_result.prob("pos")) # Output: 0.229387314312
```

```
<ProbDist with 2 samples>
neg
0.9744276603647843
0.0255723396352132
```

Result and observation of twitter data:

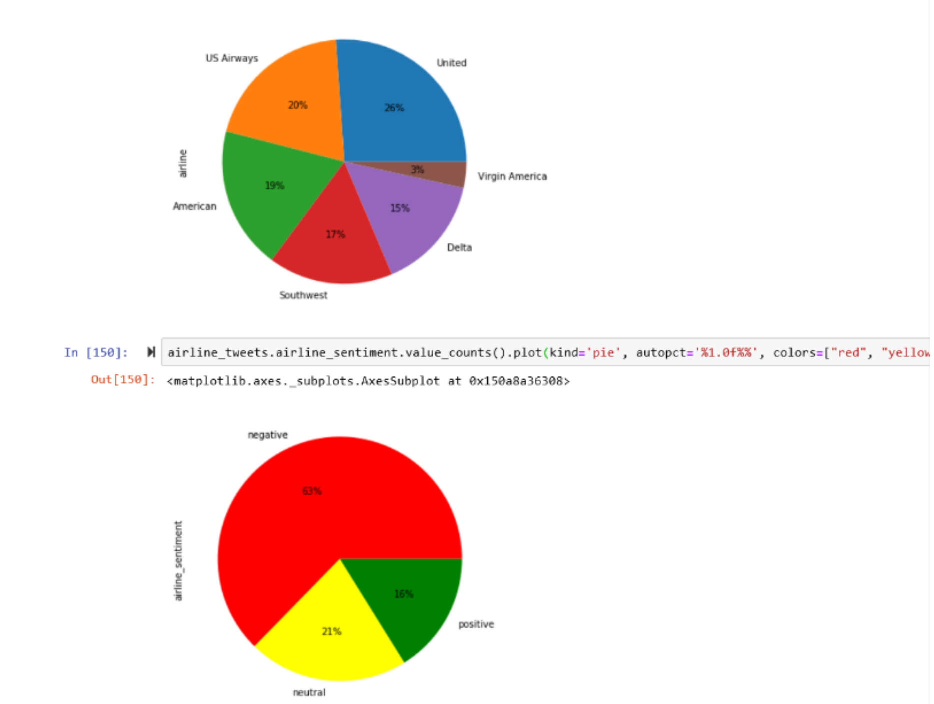
1) The result shows that the ☹ is used 2089.5 times more often in positive reviews than in negative reviews.



```
Accuracy is: 0.9953333333333333
Most Informative Features
                      :( = True           Negati : Positi =   2089.5 : 1.0
                      :) = True           Positi : Negati =    976.1 : 1.0
                     sad = True           Negati : Positi =     24.2 : 1.0
                follower = True           Positi : Negati =     20.7 : 1.0
                     bam = True           Positi : Negati =     18.7 : 1.0
               community = True           Positi : Negati =     17.4 : 1.0
                     x15 = True           Negati : Positi =     17.3 : 1.0
                    glad = True           Positi : Negati =     14.4 : 1.0
              appreciate = True           Positi : Negati =     12.8 : 1.0
                  friday = True           Positi : Negati =     12.4 : 1.0
None
```

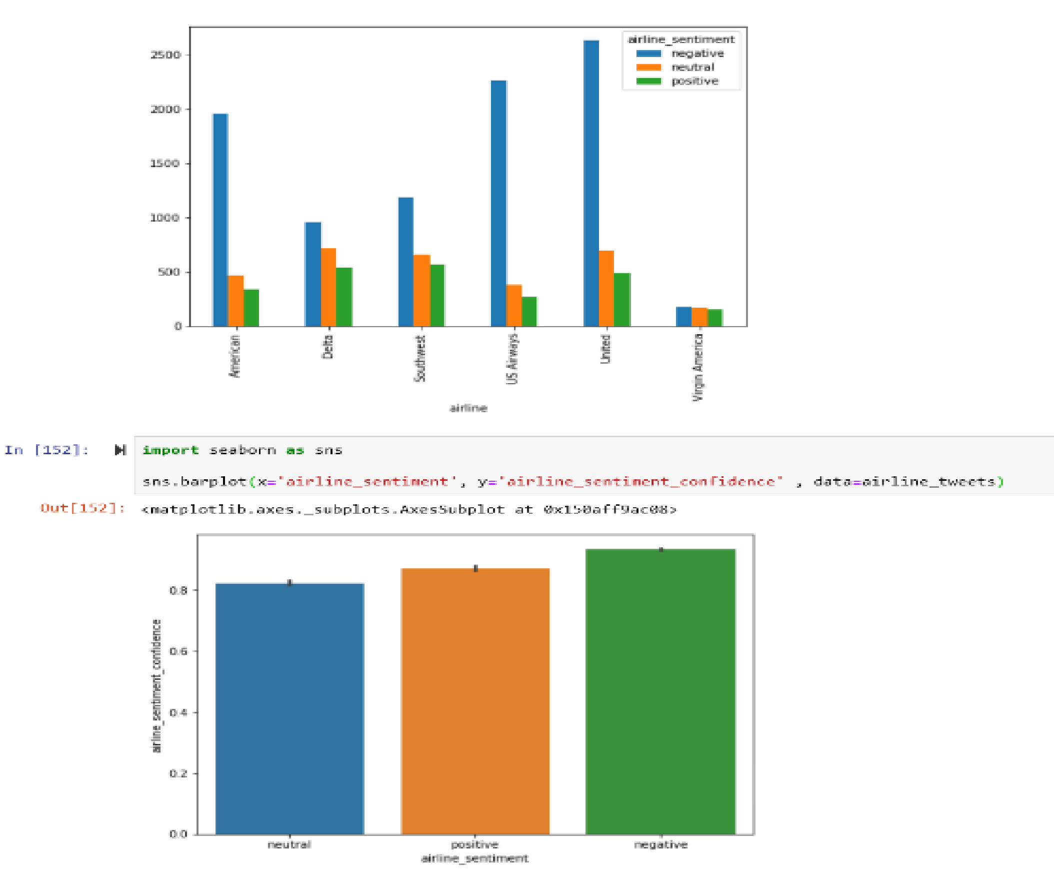
Result and observation of Airline reviews:

1) **This result shows that we have 16% positive, 21% neutral and 63% negative. It also shows the contribution of each airline.**



```
In [150]:  airline_tweets.airline_sentiment.value_counts().plot(kind='pie', autopct='%1.0f%%', colors=["red", "yellow
Out[150]:  <matplotlib.axes._subplots.AxesSubplot at 0x150a8a36308>
```

2) **This result shows the maximum negative reviews percentage: United airline has the maximum, followed by United States. Virginia has positive and negative reviews at an equal ration with a minimum contribution of 3%**



```
In [152]:   import seaborn as sns

            sns.barplot(x='airline_sentiment', y='airline_sentiment_confidence' , data=airline_tweets)
Out[152]:   <matplotlib.axes._subplots.AxesSubplot at 0x150aff9ac08>
```

Why Sentiment Analysis:

It is assessed that 80% of the world's information is unstructured, at the end of the day it is disorderly. Immense volumes of text information (messages, bolster tickets, visits, web-based life discussions, overviews, articles, records, and so forth), is made each day yet it's difficult to break down, comprehend, and sort through, also tedious and costly.

Benefits:

- ✓ Better insights,
- ✓ sorting data at a larger scale,
- ✓ real-time analysis,
- ✓ highly subjective, etc.

Challenges:

- **Subjectivity and Tone**
  Ex: The package is nice, The package is red
- **Context and Polarity**
  Ex: Everything of it.   Absolutely nothing!
- **Irony and Sarcasm**
  Ex: Yeah sure. So smooth!
  Not one, but many

- **Comparisons**
  Ex: This product is second to none
  This is better than older tool
  This is better than nothing
- **Emojis**
  Western Emojis (e.g. :D)
  Eastern emojis (e.g. ¯ \ _ (ツ) _ / ¯)
- **Defining Neutral?**

Applications:

- Social Media Monitoring
- Brand Monitoring
- Voice of Customer (VOC)
- Customer service
- Market research

One Advance feature to over come Subjectivity and tone is Tone analyser from IBM Watson tone analyzer. It helps to analyse the tone and emotions of the user when its in a written text. However, we could not explore more into it due to time restriction and complexity of codes. Our goal is to understand this by studying more about it.

Reference:

https://monkeylearn.com/sentiment-analysis/

http://blog.chapagain.com.np/python-nltk-sentiment-analysis-on-movie-reviews-natural-language-processing-nlp/

file:///C:/Users/moumi/Desktop/Busniess%20Analytucs/Sem%202/Machine%20learning/emotions.htm