Assignment 2

This assignment asks you to perform information extraction, data pre-processing, and sentiment analysis tasks on a real world data. The data contains tweets addressed to American airlines, in their original form. Data samples are provided in csv files and also output should be submitted in csv files. You may either use the provided code for reading and writing csv, or build your own data reader.

DO NOT ZIP THE SUBMISSION FILES

Joint submissions and plagiarism: This assignment is ideally intended to be an individual work. If you have worked with other students you should indicate this at the time of submission. You should indicate which questions you have worked with which other student/s. The marks for these questions will be reduced by 30%. If you do not indicate that you have worked with other students, and we still find substantial similarities in your work, it will be treated as plagiarism and handle in accordance with the university's guidelines.

Note: This assignment is largely based on Lab 8, Lab 3, Lecture 8, and Lecture 6. You may find substantial help on the NLTK website. You can either use NLTK, or your own code, or any other python library for tasks like sentence splitting, POS tagging, chunking etc.

Note: Please bear in mind that we do not expect a 100% accuracy in these tasks, but we do expect a best try.

In [3]:

```
"""this code reads the file Q1.csv and returns the contents as a list of lists, one tuple for each row in the csv file.
Modify accordingly to read the files for Q3 and Q4"""

import csv

def read_csv(inputFilePath):
    fileReader = open(inputFilePath, "r+", encoding = "utf-8", errors = "ignore")
    tweets_list = []

    for row in csv.reader(fileReader):
        list_element = [row[0],row[1]]
        tweets_list.append(list_element)

    return tweets_list


#Example usage:
```

```
"""Note: File path is relative to your python notebook directory. In the above case, the file is placed in t
he same
directory as the python notebook directory."""
print(read_csv("Assignment2.csv"))
```

```
[['681448150', '@VirginAmerica What @dhepburn said.'], ['681448158',
'@VirginAmerica it\'s really aggressive to blast obnoxious "entertain
ment" in your guests\' faces &amp;amp; they have little recourse'], ['681
448159', "@VirginAmerica and it's a really big bad thing about it"],
['681448162', "@VirginAmerica seriously would pay $30 a flight for se
ats that didn't have this playing.\nit's really the only bad thing ab
out flying VA"], ['681448171', "@VirginAmerica it was amazing, and ar
rived an hour early. You're too good to me."], ['681448176', '@Virgin
America I &amp;lt;3 pretty graphics. so much better than minimal iconogra
phy. :D'], ['681448178', "@VirginAmerica This is such a great deal! A
lready thinking about my 2nd trip to @Australia &amp;amp; I haven't even
gone on my 1st trip yet! ;p"], ['681448182', '@VirginAmerica Thanks!'
], ['681448186', "@VirginAmerica So excited for my first cross countr
y flight LAX to MCO I've heard nothing but great things about Virgin
America. #29DaysToGo"]]
```

```
In [1]:
"""Writing a list of lists to a csv file"""

import csv

def write_csv(input_list):
    csv_file = csv.writer(open("Answer1.csv", "w+"))

    for element in input_list:
        id = element[0]
        tweet = element[1]
        airline_name = element[2]
        csv_file.writerow([id, tweet, airline_name])



#Example usage:
input_list = [[681448314, "@@VirginAmerica hi I just booked a flight but need to add baggage, how c
an I do this?",
              "VirginAmerica"],
             [68144815, "@United Actually, the flight was just Cancelled Flightled! http://t.co/Qf0Oc2Hqe
Z",
              "United"]]
write_csv(input_list)
```

Question 1 (15 Marks)

Download file Q1.csv from Blackboard. Note that the first row only comprises of field names, the data values start from the second row. You can observe that the airline names (named entities) appear in certain pattern/s. Formulate regular expressions for these pattern/s.

Using regular expressions, automatically extract airline name/s from each tweet. These expressions should work for any airline name mentioned in a similar pattern, and not just the ones present in the given data. The output should be provided in a CSV file ("Answer1.csv") in the format: id, tweet, airline name/s. You can use the provided code for writing csv files.

Note: For 'id' retain the tweet id from Q1.csv. If more than one airline name is identified for a tweet, output all of them, but separated with a '' (example: VirginAmerica_Delta)

In [25]:

```
# Write your code here.
```

Question 2 (20 Marks)

Process the tweets from Q1.csv so that they resemble standard text. You can use the following pre-processing steps or any step of your liking. You can use regular expressions to do this.

- Replace 'RT' with an empty string. Example: "Beautifully smart and simple idea RT @madebymany" -> "Beautifully smart and simple idea @madebymany about our #hollegram iPad app... http://bit.ly/ieaVOB"
- Replace a sequence of punctuation marks with a single punctuation. Example: "Beautifully smart and simple idea @madebymany about our #hollegram iPad app..." -> "Beautifully smart and simple idea @madebymany about our #hollegram iPad app. http://bit.ly/ieaVOB"
- Replace url's with an empty string. (Hint: You can easily find some standard regular expressions on internet, which are used to identify URLs) Example: "Beautifully smart and simple idea @madebymany about our #hollegram iPad app. http://bit.ly/ieaVOB" -> "Beautifully smart and simple idea @madebymany about our #hollegram iPad app."
- Replace @username with username. Example:"Beautifully smart and simple idea @madebymany about our #hollegram iPad app." -> Beautifully smart and simple idea madebymany about our #hollegram iPad app.
- Replace hashtags with an empty string if it appears at the end of the sentence, else only remove the # symbol. Example: "Beautifully smart and simple idea madebymany about our #hollegram iPad app." -> "Beautifully smart and simple idea madebymany about our hollegram iPad app."

Now, use NLTKs NER to identify all the Named Entities in the processed tweets. Similar to Q1, add the named entities identified in this step to a csv file ("Answer2.csv") in the format: id, tweet, named_entity.

In [ ]:

```
# Write your code here.
```

Question 3 (25 Marks)

Now, build a rule based (non-statistical) sentiment classifier for tweets, which predicts a given tweet's sentiment towards a given airline. You can use the pre-processing steps from Q2, but no marks would be deducted if you don't. Use the following steps to build the classifier:

1. Split the tweets into individual sentences. Perform POS tagging of these sentences.
2. Identify all the adjectives in the tweet.
3. Obtain an sentiment value for those adjectives using SentiWordnet (you can use the provided code or your own code). This step calculates a sentiment value for each adjective.
4. Add all the sentiment values for all the adjective for a tweet.
5. The prediced sentiment is 'negative' if the sum in step 4 is a negative value, 'positive' if positive value, and 'neutral' if the sum is 0.

Calculate the Precision, Recall and F measure for this sentiment classifier using the given test dataset (Q3.csv).

In [ ]:

```
import nltk
nltk.download()

"""When executed it launces a GUI that allows you to select and download nltk resources.
Go to the tab "Corpora". Find SentiWordNet in the displayed list, select it, and then click "Download".
Wait for it to
be downloaded"""
```

In [6]:

```
from nltk.corpus import sentiwordnet as swn

"""This method returns an approximate sentiment value for each entry of the input adjective list,
a negative value is returned for negative sentiment"""
def sentiwordnet_lookup(adjective_list):
    adj_sentiment = []
    for adj in adjective_list:
        adj_synsets = swn.senti_synsets(adj,'a')
        try:
            adj0 = (list(adj_synsets))[0]
        except IndexError:
            adj0 = 'null'
        if(adj0 == 'null'):
```

```
            adj_sentiment_entry = [adj, "not found"]
            adj_sentiment.append(adj_sentiment_entry)
        else:
            adj0_senti_scores = [adj0.pos_score(), adj0.neg_score(), adj0.obj_score()]
            adj0_senti_score = max(adj0_senti_scores)
            if(adj0_senti_score==adj0.neg_score()):
                adj0_senti_score = adj0_senti_score*(-1)
            elif(adj0_senti_score==adj0.obj_score()):
                adj0_senti_score = 0

            adj_sentiment_entry = [adj, adj0_senti_score]

            adj_sentiment.append(adj_sentiment_entry)

    return adj_sentiment

#Example usage:
print(sentiwordnet_lookup(["bad","obnoxious"]))
```

```
[['bad', -0.625], ['obnoxious', 'not found']]
```

In [ ]:

```
# Write your code here.
```

Question 4 (25 Marks)

Provided a training dataset of tweets (Q4.csv), build a SVM classifier (using sklearn) to predict the tweet sentiment towards a given airline. Use words as features. Calculate Precision, Recall and F-measure using Q3.csv as test data.

_Note: You can take hints from the Assignment 1 Task 5.

_Note: You can use the pre-processing steps from Q2 in order to clean the tweets before training the classifier, as well as before feeding in the test data to the trained model. This is likely to improve the classification accuracy. No marks would be deducted if you don't perform this step.

In [ ]:

```
# Write your code here.
```

Question 5 (15 Marks)

a)Which approach seems to work better for identifying airline names (Q1 or Q2)? Why do you think it worked better?
Answer...
b) Do you think the approach in Q1 would work with any given text (tweet or non-tweet). Why?

Answer...

c)Which approach produced better F-score for sentiment prediction (Q3 or Q4)? And why do you think it worked better?

Answer...