Configuration

In [ ]:

```
# Parameters
PROJECT_NAME = 'ML1010_Weekly'
ENABLE_COLAB = True

#Root Machine Learning Directory. Projects appear underneath
GOOGLE_DRIVE_MOUNT = '/content/gdrive'
COLAB_ROOT_DIR = GOOGLE_DRIVE_MOUNT + '/MyDrive/Colab Notebooks'
COLAB_INIT_DIR = COLAB_ROOT_DIR + '/utility_files'

LOCAL_ROOT_DIR = '/home/magni/Documents/ML_Projects'
LOCAL_INIT_DIR = LOCAL_ROOT_DIR + '/utility_files'
```

Bootstrap Environment

In [ ]:

```
#add in support for utility file directory and importing
import sys
import os

if ENABLE_COLAB:
  #Need access to drive
  from google.colab import drive
  drive.mount(GOOGLE_DRIVE_MOUNT, force_remount=True)

  #add in utility directory to syspath to import
  INIT_DIR = COLAB_INIT_DIR
  sys.path.append(os.path.abspath(INIT_DIR))

  #Config environment variables
  ROOT_DIR = COLAB_ROOT_DIR

else:
  #add in utility directory to syspath to import
  INIT_DIR = LOCAL_INIT_DIR
  sys.path.append(os.path.abspath(INIT_DIR))

  #Config environment variables
  ROOT_DIR = LOCAL_ROOT_DIR

#Import Utility Support
from jarvis import Jarvis
jarvis = Jarvis(ROOT_DIR, PROJECT_NAME)

import mv_python_utils as mvutils
```

```
Mounted at /content/gdrive
Wha...where am I?
I am awake now.

I have set your current working directory to /content/gdrive/MyDrive/Colab No
tebooks/ML1010_Weekly
```

```
The current time is 19:35
Hello sir. I hope you had dinner.
```

Emotion and Sentiment Analysis

Sentiment analysis is perhaps one of the most popular applications of NLP, with a vast number of tutorials, courses, and applications that focus on analyzing sentiments of diverse datasets ranging from corporate surveys to movie reviews. The key aspect of sentiment analysis is to analyze a body of text for understanding the opinion expressed by it. Typically, we quantify this sentiment with a positive or negative value, called polarity. The overall sentiment is often inferred as positive, neutral or negative from the sign of the polarity score.

Usually, sentiment analysis works best on text that has a subjective context than on text with only an objective context. Objective text usually depicts some normal statements or facts without expressing any emotion, feelings, or mood. Subjective text contains text that is usually expressed by a human having typical moods, emotions, and feelings. Sentiment analysis is widely used, especially as a part of social media analysis for any domain, be it a business, a recent movie, or a product launch, to understand its reception by the people and what they think of it based on their opinions or, you guessed it, sentiment!

Typically, sentiment analysis for text data can be computed on several levels, including on an individual sentence level, paragraph level, or the entire document as a whole. Often, sentiment is computed on the document as a whole or some aggregations are done after computing the sentiment for individual sentences. There are two major approaches to sentiment analysis.

- Supervised machine learning or deep learning approaches
- Unsupervised lexicon-based approaches

For the first approach we typically need pre-labeled data. Hence, we will be focusing on the second approach. For a comprehensive coverage of sentiment analysis, refer to Chapter 7: Analyzing Movie Reviews Sentiment, Practical Machine Learning with Python, Springer\Apress, 2018. In this scenario, we do not have the convenience of a well-labeled training dataset. Hence, we will need to use unsupervised techniques for predicting the sentiment by using knowledgebases, ontologies, databases, and lexicons that have detailed information, specially curated and prepared just for sentiment analysis. A lexicon is a dictionary, vocabulary, or a book of words. In our case, lexicons are special dictionaries or vocabularies that have been created for analyzing sentiments. Most of these lexicons have a list of positive and negative polar words with some score associated with them, and using various techniques like the position of words, surrounding words, context, parts of speech, phrases, and so on, scores are assigned to the text documents for which we want to compute the sentiment. After aggregating these scores, we get the final sentiment.

Various popular lexicons are used for sentiment analysis, including the following.

AFINN lexicon Bing Liu's lexicon MPQA subjectivity lexicon SentiWordNet VADER lexicon TextBlob lexicon This is not an exhaustive list of lexicons that can be leveraged for sentiment analysis, and there are several other lexicons which can be easily obtained from the Internet. Feel free to check out each of these links and explore them. We will be covering two techniques in this section

Some Pre-Processing

Import necessary depencencies

In [ ]:
```
import pandas as pd
import numpy as np
#import model_evaluation_utils as meu

np.set_printoptions(precision=2, linewidth=80)
```

In [ ]:
```
!pip install Afinn
```

```
Collecting Afinn
  Downloading afinn-0.1.tar.gz (52 kB)
     |████████████████████████████████| 52 kB 366 kB/s
Building wheels for collected packages: Afinn
  Building wheel for Afinn (setup.py) ... done
  Created wheel for Afinn: filename=afinn-0.1-py3-none-any.whl size=53448 sha
256=c3f0ed2f6827bfc678d09a0b9e8313652b56fd542d69d7dc5d640a0f23e220e6
  Stored in directory: /root/.cache/pip/wheels/9d/16/3a/9f0953027434eab5dadf3
f33ab3298fa95afa8292fcf7aba75
Successfully built Afinn
Installing collected packages: Afinn
Successfully installed Afinn-0.1
```

Load and normalize data

1. Cleaning Text - strip HTML
2. Removing accented characters
3. Expanding Contractions
4. Removing Special Characters
5. Lemmatizing text¶
6. Removing Stopwords

In [ ]:
```
dataset = pd.read_csv(jarvis.DATA_DIR + '/movie_reviews_cleaned.csv')

reviews = np.array(dataset['review'])
sentiments = np.array(dataset['sentiment'])

# extract data for model evaluation
train_reviews = reviews[:35000]
train_sentiments = sentiments[:35000]

test_reviews = reviews[35000:]
test_sentiments = sentiments[35000:]
sample_review_ids = [7626, 3533, 13010]
```

In [ ]:
```
# SKIP FOR THE STUDENTS BECAUSE INSTRUCTOR HAS PRE_NORMALIZED AND SAVED THE F
# normalize dataset (time consuming using spacey pipeline)
"""
norm_test_reviews = tn.normalize_corpus(test_reviews)
norm_train_reviews = tn.normalize_corpus(train_reviews)
#output back to a csv file again
import csv
with open(r'movie_reviews_cleaned.csv', mode='w') as cleaned_file:
    csv_writer = csv.writer(cleaned_file, delimiter=',', quotechar='"', quoti
    csv_writer.writerow(['review', 'sentiment'])
    for  text, sent in zip(norm_test_reviews, test_sentiments):
        csv_writer.writerow([text, sent])
    for  text, sent in zip(norm_train_reviews, train_sentiments):
        csv_writer.writerow([text, sent])
"""
```

Out[ ]:
```
'\nnorm_test_reviews = tn.normalize_corpus(test_reviews)\nnorm_train_reviews
= tn.normalize_corpus(train_reviews)\n#output back to a csv file again\nimpor
t csv\nwith open(r\'movie_reviews_cleaned.csv\', mode=\'w\') as cleaned_fil
e:\n    csv_writer = csv.writer(cleaned_file, delimiter=\',\', quotechar
=\'"\', quoting=csv.QUOTE_MINIMAL)\n    csv_writer.writerow([\'review\', \'se
ntiment\'])\n    for  text, sent in zip(norm_test_reviews, test_sentiments):\
n        csv_writer.writerow([text, sent])\n    for  text, sent in zip(norm_t
rain_reviews, train_sentiments):\n        csv_writer.writerow([text, sent])\n
'
```

==========

Part A. Unsupervised (Lexicon) Sentiment Analysis

==========

1. Sentiment Analysis with AFINN

The AFINN lexicon is perhaps one of the simplest and most popular lexicons that can be used extensively for sentiment analysis. Developed and curated by Finn Arup Nielsen, you can find more details on this lexicon in the paper, "A new ANEW: evaluation of a word list for sentiment analysis in microblogs", proceedings of the ESWC 2011 Workshop. The current version of the lexicon is AFINN-en-165. txt and it contains over 3,300+ words with a polarity score associated with each word. You can find this lexicon at the author's official GitHub repository along with previous versions of it, including AFINN-111. The author has also created a nice wrapper library on top of this in Python called afinn, which we will be using for our analysis

In [ ]:
```
from afinn import Afinn

afn = Afinn(emoticons=True)

# NOTE:  to use afinn score, call the function afn.score("text you want the s
# the lexicon will be used to compute summary of sentiment for the given text
```

Predict sentiment for sample reviews

We can get a good idea of general sentiment for different sample.

In [ ]:
```
for review, sentiment in zip(test_reviews[sample_review_ids], test_sentiments
    print('REVIEW:', review)
    print('Actual Sentiment:', sentiment)
    print('Predicted Sentiment polarity:', afn.score(review))
    print('-'*60)
```

```
REVIEW: word fail whenever want describe feeling movie sequel flaw sure start
subspecie not execute well enough special effect glorify movie herd movie mas
s consumer care quantity quality cheap fun depth crap like blade not even des
erve capital letter underworlddracula 2000dracula 3000 good movie munch popco
rn drink couple coke make subspecie superior effort anyone claim vampire fana
tic hand obvious vampire romanian story set transylvania scene film location
convince atmosphere not base action pack chase expensive orchestral music rad
u source atmosphere vampire look like behave add breathtakingly gloomy castle
dark passageway situate romania include typical vampiric element movement sha
dow wall vampire take flight work art short like fascinated vampire feel appe
arance well setting sinister dark no good place look subspecie movie vampire
journal brilliant spin former
Actual Sentiment: positive
Predicted Sentiment polarity: 20.0
------------------------------------------------------------
REVIEW: good family movie laugh wish not much school stuff like bully fill mo
vie also seem little easy save piece land build mean flow easily make aware w
ildlife cute way introduce piece land fast runner little slow little hokey re
mind go back school oh dvd chock full goody not miss 7 10 movie 10 10 dvd ext
ra well worth watch well worth time see
Actual Sentiment: positive
Predicted Sentiment polarity: 12.0
------------------------------------------------------------
REVIEW: opinion movie not good hardly find good thing say still would like ex
plain conclude another bad movie decide watch costas mandylor star main reaso
n watch till end like action movie understand movie build action rather story
know not go detail come credibility story event even not explain scene lack s
ense reality look ridiculous beginning movie look quite promising tough good
```

```
look specialist not tough smart funny partner must job turn bit different exp
ect story take place cruise ship disaster happen ship turn leave alive strugg
le survive escape shark professional killer rise water furthermore movie quit
e violent main weapon beside disaster already take passenger gun successfully
use many case personally miss good man man woman woman prefer fight family fu
n not think think movie shoot hurry without real vision try say make usual ac
tion movie trick bit something call love without real meaning result bad movi
e
Actual Sentiment: negative
Predicted Sentiment polarity: 2.0
------------------------------------------------------------
```

Predict sentiment for test dataset

In [ ]:
```
sentiment_polarity = [afn.score(review) for review in test_reviews]
predicted_sentiments = ['positive' if score >= 1.0 else 'negative' for score
```

In [ ]:
```
display(type(sentiment_polarity))
print(sentiment_polarity[4])
```

```
list
12.0
```

Evaluate model performance

In [ ]:
```
from sklearn import metrics

results = metrics.classification_report(test_sentiments, predicted_sentiments
print(results)

#meu.display_model_performance_metrics(true_labels=test_sentiments, predicted
#                                      classes=['positive', 'negative'])
```

```
              precision    recall  f1-score   support

    negative       0.78      0.56      0.65      7413
    positive       0.66      0.84      0.74      7587

    accuracy                           0.71     15000
   macro avg       0.72      0.70      0.70     15000
weighted avg       0.72      0.71      0.70     15000
```

2. Sentiment Analysis with SentiWordNet

SentiWordNet is a lexical resource for opinion mining. SentiWordNet assigns to each synset of WordNet three sentiment scores: positivity, negativity, objectivity. SentiWordNet is described in details in the papers:

In [ ]:

```
from nltk.corpus import sentiwordnet as swn
import nltk
nltk.download('sentiwordnet')

awesome = list(swn.senti_synsets('awesome', 'a'))[0]
print('Positive Polarity Score:', awesome.pos_score())
print('Negative Polarity Score:', awesome.neg_score())
print('Objective Score:', awesome.obj_score())
```

```
[nltk_data] Downloading package sentiwordnet to
[nltk_data]     /home/anniee/nltk_data...
[nltk_data]   Package sentiwordnet is already up-to-date!
Positive Polarity Score: 0.875
Negative Polarity Score: 0.125
Objective Score: 0.0
```

Build model

For each word in the review, add up the sentiment score of words that are NN, VB, JJ, RB if it's in the lexicon dictionary.

In [ ]:
```
import text_normalizer as tn

def analyze_sentiment_sentiwordnet_lexicon(review,
                                           verbose=False):

    # tokenize and POS tag text tokens
    tagged_text = [(token.text, token.tag_) for token in tn.nlp(review)]
    pos_score = neg_score = token_count = obj_score = 0
    # get wordnet synsets based on POS tags
    # get sentiment scores if synsets are found
    for word, tag in tagged_text:
        ss_set = None
        if 'NN' in tag and list(swn.senti_synsets(word, 'n')):
            ss_set = list(swn.senti_synsets(word, 'n'))[0]
        elif 'VB' in tag and list(swn.senti_synsets(word, 'v')):
            ss_set = list(swn.senti_synsets(word, 'v'))[0]
        elif 'JJ' in tag and list(swn.senti_synsets(word, 'a')):
            ss_set = list(swn.senti_synsets(word, 'a'))[0]
        elif 'RB' in tag and list(swn.senti_synsets(word, 'r')):
            ss_set = list(swn.senti_synsets(word, 'r'))[0]
        # if senti-synset is found
        if ss_set:
            # add scores for all found synsets
            pos_score += ss_set.pos_score()
            neg_score += ss_set.neg_score()
            obj_score += ss_set.obj_score()
            token_count += 1

    # aggregate final scores
    final_score = pos_score - neg_score
    norm_final_score = round(float(final_score) / token_count, 2)
    final_sentiment = 'positive' if norm_final_score >= 0 else 'negative'
    if verbose:
        norm_obj_score = round(float(obj_score) / token_count, 2)
        norm_pos_score = round(float(pos_score) / token_count, 2)
        norm_neg_score = round(float(neg_score) / token_count, 2)
        # to display results in a nice table
        sentiment_frame = pd.DataFrame([[final_sentiment, norm_obj_score, nor
                                         norm_neg_score, norm_final_score]],
                                       columns=pd.MultiIndex(levels=[['SENTIM
                                                                      ['Predicted Sent
                                                                       'Positive', 'Ne
                                                                      labels=[[0,0,0,0
        print(sentiment_frame)

    return final_sentiment
```

Predict sentiment for sample reviews

In [ ]:
```
for review, sentiment in zip(test_reviews[sample_review_ids], test_sentiments
    print('REVIEW:', review)
    print('Actual Sentiment:', sentiment)
    pred = analyze_sentiment_sentiwordnet_lexicon(review, verbose=True)
    print('-'*60)
```

```
REVIEW: word fail whenever want describe feeling movie sequel flaw sure start
```

```
subspecie not execute well enough special effect glorify movie herd movie mas
s consumer care quantity quality cheap fun depth crap like blade not even des
erve capital letter underworlddracula 2000dracula 3000 good movie munch popco
rn drink couple coke make subspecie superior effort anyone claim vampire fana
tic hand obvious vampire romanian story set transylvania scene film location
convince atmosphere not base action pack chase expensive orchestral music rad
u source atmosphere vampire look like behave add breathtakingly gloomy castle
dark passageway situate romania include typical vampiric element movement sha
dow wall vampire take flight work art short like fascinated vampire feel appe
arance well setting sinister dark no good place look subspecie movie vampire
journal brilliant spin former
Actual Sentiment: positive
     SENTIMENT STATS:
  Predicted Sentiment Objectivity Positive Negative Overall
0            positive        0.84     0.09     0.06    0.03
------------------------------------------------------------
REVIEW: good family movie laugh wish not much school stuff like bully fill mo
vie also seem little easy save piece land build mean flow easily make aware w
ildlife cute way introduce piece land fast runner little slow little hokey re
mind go back school oh dvd chock full goody not miss 7 10 movie 10 10 dvd ext
ra well worth watch well worth time see
Actual Sentiment: positive
     SENTIMENT STATS:
  Predicted Sentiment Objectivity Positive Negative Overall
0            positive        0.85     0.08     0.06    0.02
------------------------------------------------------------
REVIEW: opinion movie not good hardly find good thing say still would like ex
plain conclude another bad movie decide watch costas mandylor star main reaso
n watch till end like action movie understand movie build action rather story
know not go detail come credibility story event even not explain scene lack s
ense reality look ridiculous beginning movie look quite promising tough good
look specialist not tough smart funny partner must job turn bit different exp
ect story take place cruise ship disaster happen ship turn leave alive strugg
le survive escape shark professional killer rise water furthermore movie quit
e violent main weapon beside disaster already take passenger gun successfully
use many case personally miss good man man woman woman prefer fight family fu
n not think think movie shoot hurry without real vision try say make usual ac
tion movie trick bit something call love without real meaning result bad movi
e
Actual Sentiment: negative
     SENTIMENT STATS:
  Predicted Sentiment Objectivity Positive Negative Overall
0            positive        0.82     0.09     0.09    -0.0
------------------------------------------------------------
```

Predict sentiment for test dataset

In [ ]:

```
predicted_sentiments = [analyze_sentiment_sentiwordnet_lexicon(review, verbos
```

Evaluate model performance

In [ ]:

```
results = metrics.classification_report(test_sentiments, predicted_sentiments
print(results)
```

```
              precision    recall  f1-score   support
```

```
    negative       0.71      0.60      0.65      7413
    positive       0.66      0.76      0.71      7587

   micro avg       0.68      0.68      0.68     15000
   macro avg       0.69      0.68      0.68     15000
weighted avg       0.69      0.68      0.68     15000
```

3. Sentiment Analysis with VADER

In [ ]:
```
from nltk.sentiment.vader import SentimentIntensityAnalyzer
```

```
/home/anniee/.local/lib/python3.6/site-packages/nltk/twitter/__init__.py:20:
UserWarning: The twython library has not been installed. Some functionality f
rom the twitter package will not be available.
  warnings.warn("The twython library has not been installed. "
```

Build model

In [ ]:
```
def analyze_sentiment_vader_lexicon(review,
                                    threshold=0.1,
                                    verbose=False):
    # pre-process text
    review = tn.strip_html_tags(review)
    review = tn.remove_accented_chars(review)
    review = tn.expand_contractions(review)

    # analyze the sentiment for review
    analyzer = SentimentIntensityAnalyzer()
    scores = analyzer.polarity_scores(review)
    # get aggregate scores and final sentiment
    agg_score = scores['compound']
    final_sentiment = 'positive' if agg_score >= threshold\
                                   else 'negative'
    if verbose:
        # display detailed sentiment statistics
        positive = str(round(scores['pos'], 2)*100)+'%'
        final = round(agg_score, 2)
        negative = str(round(scores['neg'], 2)*100)+'%'
        neutral = str(round(scores['neu'], 2)*100)+'%'
        sentiment_frame = pd.DataFrame([[final_sentiment, final, positive,
                                         negative, neutral]],
                                         columns=pd.MultiIndex(levels=[['SENTI
                                                                        ['Predi
                                                                         'Posit
                                                                labels=[[0,0,0,

        print(sentiment_frame)

    return final_sentiment
```

Predict sentiment for sample reviews

In [ ]:

```
nltk.download('vader_lexicon')

for review, sentiment in zip(test_reviews[sample_review_ids], test_sentiments
    print('REVIEW:', review)
    print('Actual Sentiment:', sentiment)
    pred = analyze_sentiment_vader_lexicon(review, threshold=0.4, verbose=Tru
    print('-'*60)
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     /home/anniee/nltk_data...
REVIEW: word fail whenever want describe feeling movie sequel flaw sure start
subspecie not execute well enough special effect glorify movie herd movie mas
s consumer care quantity quality cheap fun depth crap like blade not even des
erve capital letter underworlddracula 2000dracula 3000 good movie munch popco
rn drink couple coke make subspecie superior effort anyone claim vampire fana
tic hand obvious vampire romanian story set transylvania scene film location
convince atmosphere not base action pack chase expensive orchestral music rad
u source atmosphere vampire look like behave add breathtakingly gloomy castle
dark passageway situate romania include typical vampiric element movement sha
dow wall vampire take flight work art short like fascinated vampire feel appe
arance well setting sinister dark no good place look subspecie movie vampire
journal brilliant spin former
Actual Sentiment: positive
     SENTIMENT STATS:
  Predicted Sentiment Polarity Score             Positive Negative Neutral
0            positive           0.98  28.000000000000004%    11.0%   61.0%
------------------------------------------------------------
REVIEW: good family movie laugh wish not much school stuff like bully fill mo
vie also seem little easy save piece land build mean flow easily make aware w
ildlife cute way introduce piece land fast runner little slow little hokey re
mind go back school oh dvd chock full goody not miss 7 10 movie 10 10 dvd ext
ra well worth watch well worth time see
Actual Sentiment: positive
     SENTIMENT STATS:
  Predicted Sentiment Polarity Score Positive Negative             Neutral
0            positive           0.97    39.0%     4.0%  57.99999999999999%
------------------------------------------------------------
REVIEW: opinion movie not good hardly find good thing say still would like ex
plain conclude another bad movie decide watch costas mandylor star main reaso
n watch till end like action movie understand movie build action rather story
know not go detail come credibility story event even not explain scene lack s
ense reality look ridiculous beginning movie look quite promising tough good
look specialist not tough smart funny partner must job turn bit different exp
ect story take place cruise ship disaster happen ship turn leave alive strugg
le survive escape shark professional killer rise water furthermore movie quit
e violent main weapon beside disaster already take passenger gun successfully
use many case personally miss good man man woman woman prefer fight family fu
n not think think movie shoot hurry without real vision try say make usual ac
tion movie trick bit something call love without real meaning result bad movi
e
Actual Sentiment: negative
     SENTIMENT STATS:
  Predicted Sentiment Polarity Score Positive Negative             Neutral
0            negative          -0.98    12.0%    31.0%  56.00000000000001%
------------------------------------------------------------
```

Predict sentiment for test dataset

In [ ]:
```
predicted_sentiments = [analyze_sentiment_vader_lexicon(review, threshold=0.4
```

Evaluate model performance

In [ ]:
```
display_model_performance_metrics(true_labels=test_sentiments, predicted_labe
                                  classes=['positive', 'negative'])
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-5c81cb959a93> in <module>()
----> 1 display_model_performance_metrics(true_labels=test_sentiments, predic
ted_labels=predicted_sentiments,
      2                                   classes=['positive', 'negative'])

NameError: name 'display_model_performance_metrics' is not defined
```

In [ ]:
```

```