**Sentiment Analysis**

**Introduction**

So far, all of the analysis we've done has been pretty generic - looking at counts, creating scatter plots, etc. These techniques could be applied to numeric data as well.

When it comes to text data, there are a few popular techniques that we'll be going through in the next few notebooks, starting with sentiment analysis. A few key points to remember with sentiment analysis.

1. **TextBlob Module:** Linguistic researchers have labeled the sentiment of words based on their domain expertise. Sentiment of words can vary based on where it is in a sentence. The TextBlob module allows us to take advantage of these labels.
2. **Sentiment Labels:** Each word in a corpus is labeled in terms of polarity and subjectivity (there are more labels as well, but we're going to ignore them for now). A corpus' sentiment is the average of these.
   * **Polarity**: How positive or negative a word is. -1 is very negative. +1 is very positive.
   * **Subjectivity**: How subjective, or opinionated a word is. 0 is fact. +1 is very much an opinion. (meaning of subjectivity is “the quality of being based on or influenced by personal feelings, tastes, or opinions.”” the quality of existing in someone's mind rather than the external world.”)

For more info on how TextBlob coded up its [sentiment function](https://planspace.org/20150607-textblob_sentiment/).

Let's take a look at the sentiment of the various transcripts, both overall and throughout the comedy routine.

**Sentiment of Routine**

*# We'll start by reading in the corpus, which preserves word order*

**import** **pandas** **as** **pd**

data = pd.read\_pickle('corpus.pkl')

data

*# Create quick lambda functions to find the polarity and subjectivity of each routine*

*# Terminal / Anaconda Navigator: conda install -c conda-forge textblob*

**from** **textblob** **import** TextBlob

pol = **lambda** x: TextBlob(x).sentiment.polarity

sub = **lambda** x: TextBlob(x).sentiment.subjectivity

data['polarity'] = data['transcript'].apply(pol)

data['subjectivity'] = data['transcript'].apply(sub)

data

*# Let's plot the results*

**import** **matplotlib.pyplot** **as** **plt**

plt.rcParams['figure.figsize'] = [10, 8]

**for** index, comedian **in** enumerate(data.index):

x = data.polarity.loc[comedian]

y = data.subjectivity.loc[comedian]

plt.scatter(x, y, color='blue')

plt.text(x+.001, y+.001, data['full\_name'][index], fontsize=10)

plt.xlim(-.01, .12)

plt.title('Sentiment Analysis', fontsize=20)

plt.xlabel('<-- Negative -------- Positive -->', fontsize=15)

plt.ylabel('<-- Facts -------- Opinions -->', fontsize=15)

plt.show()

## Sentiment of Routine Over Time

Instead of looking at the overall sentiment, let's see if there's anything interesting about the sentiment over time throughout each routine.

*# Split each routine into 10 parts*

**import** **numpy** **as** **np**

**import** **math**

**def** split\_text(text, n=10):

*'''Takes in a string of text and splits into n equal parts, with a default of 10 equal parts.'''*

*# Calculate length of text, the size of each chunk of text and the starting points of each chunk of text*

length = len(text)

size = math.floor(length / n)

start = np.arange(0, length, size)

*# Pull out equally sized pieces of text and put it into a list*

split\_list = []

**for** piece **in** range(n):

split\_list.append(text[start[piece]:start[piece]+size])

**return** split\_list

*# Let's take a look at our data again*

data

*# Let's create a list to hold all of the pieces of text*

list\_pieces = []

**for** t **in** data.transcript:

split = split\_text(t)

list\_pieces.append(split)

list\_pieces

*# The list has 10 elements, one for each transcript*

len(list\_pieces)

*# Each transcript has been split into 10 pieces of text*

len(list\_pieces[0])

*# Calculate the polarity for each piece of text*

polarity\_transcript = []

**for** lp **in** list\_pieces:

polarity\_piece = []

**for** p **in** lp:

polarity\_piece.append(TextBlob(p).sentiment.polarity)

polarity\_transcript.append(polarity\_piece)

polarity\_transcript

*# Show the plot for one comedian*

plt.plot(polarity\_transcript[0])

plt.title(data['full\_name'].index[0])

plt.show()

*# Show the plot for all comedians*

plt.rcParams['figure.figsize'] = [16, 12]

**for** index, comedian **in** enumerate(data.index):

plt.subplot(3, 4, index+1)

plt.plot(polarity\_transcript[index])

plt.plot(np.arange(0,10), np.zeros(10))

plt.title(data['full\_name'][index])

plt.ylim(ymin=-.2, ymax=.3)

plt.show()

Ali Wong stays generally positive throughout her routine. Similar comedians are Louis C.K. and Mike Birbiglia.

On the other hand, you have some pretty different patterns here like Bo Burnham who gets happier as time passes and Dave Chappelle who has some pretty down moments in his routine.

**Additional Exercises**

1. Modify the number of sections the comedy routine is split into and see how the charts over time change.