CS156 Final assignment

Natural language processing with comic transcript

Overview

In this assignment, I'm classifying sentiments from comic transcripts with machine learning model.

Data

The comic transcripts are from https://scrapsfromtheloft.com/. There are 10 comedians with 10 transcripts from their Netflix special. I divided each script into 200 smaller equal pieces of text, and labeled them into "Positive", "Neutral", and "Negative" sentiment using TextBlob.

I could have divided the script into individual sentences, but that would result in (1) unequal data training from each comedian and (2) more computational resource requirement in training the model.

Approach

1. I scraped the data from the website, did some text processing and data exploration.
2. Then I tokenized the scripts into bag of words that denotes the vocabulary used by the comics. This was done by either Count Vectorizer or TF-IDF Vectorizer.
3. I used Neural Network (for only Count Vectorizer) and Logistic Regression (for both Count Vectorizer and TF-IDF Vectorizer) to classify the data.

Result

Here are the performance of the model:

- Neural Network: 64.17% (with Count Vectorizer)
- Logistic Regression: 62% (with Count Vectorizer) and 61.5% (with TF-IDF Vectorizer)

It seems like both models have comparable performance, around 60%, but they are heavily overfitted because the training data (with Count Vectorizer) for both models result in 100% accuracy. There must be some way to prevent overfitting (like doing image augmentation with images data), but as I'm not super familiar with text data, I was not able to prevent overfitting.

I also couldn't use TF-IDF Vectorizer to work with Neural Network because of some formatting issue. Otherwise, I would have been a meaningful comparison.

References

The code is adapted from these 2 tutorials: Natural language processing in Python, and A guide to text classification and sentiment analysis

Getting the data

```
# Data scraping
from bs4 import BeautifulSoup
import requests
import pickle

# Data cleaning
import pandas as pd
import re
import string
```

```
# Function to scrape data
def url_to_transcript(url):
    '''Scrape transcript data from URL'''
    page = requests.get(url).text
    soup = BeautifulSoup(page, "lxml")
    text = [p.text for p in soup.find(class_="site-content").find_all('p')]
    print(f"Getting data from: {url}")
    return text
```

```
# URLs of transcripts
urls = ['http://scrapsfromtheloft.com/2017/09/19/ali-wong-baby-cobra-2016-full-tra
        'http://scrapsfromtheloft.com/2017/08/03/anthony-jeselnik-thoughts-prayers-
        'http://scrapsfromtheloft.com/2017/05/24/bill-burr-im-sorry-feel-way-2014-f
        'http://scrapsfromtheloft.com/2017/04/11/dave-chappelle-age-spin-2017-full-
        'http://scrapsfromtheloft.com/2017/04/21/jim-jefferies-bare-2014-full-trans
        'http://scrapsfromtheloft.com/2017/08/19/joe-rogan-triggered-2016-full-tra
        'http://scrapsfromtheloft.com/2017/08/02/john-mulaney-comeback-kid-2015-ful
        'http://scrapsfromtheloft.com/2017/05/06/louis-ck-oh-my-god-full-transcript
        'http://scrapsfromtheloft.com/2018/03/03/mike-birbiglia-my-girlfriends-boyf
        'http://scrapsfromtheloft.com/2018/03/15/ricky-gervais-humanity-transcript/

# Scrape the transcripts and put them into a list
transcripts = [url_to_transcript(u) for u in urls]
```

```
Getting data from: http://scrapsfromtheloft.com/2017/09/19/ali-wong-baby-cobra-2(
Getting data from: http://scrapsfromtheloft.com/2017/08/03/anthony-jeselnik-thou(
Getting data from: http://scrapsfromtheloft.com/2017/05/24/bill-burr-im-sorry-fe(
Getting data from: http://scrapsfromtheloft.com/2017/04/11/dave-chappelle-age-sp(
Getting data from: http://scrapsfromtheloft.com/2017/04/21/jim-jefferies-bare-20(
Getting data from: http://scrapsfromtheloft.com/2017/08/19/joe-rogan-triggered-2(
Getting data from: http://scrapsfromtheloft.com/2017/08/02/john-mulaney-comeback-
```

```
Getting data from: http://scrapsfromtheloft.com/2017/05/06/louis-ck-oh-my-god-fu
Getting data from: http://scrapsfromtheloft.com/2018/03/03/mike-birbiglia-my-gir
Getting data from: http://scrapsfromtheloft.com/2018/03/15/ricky-gervais-humanit
```

```
# Organize data according to comedians
comedians = ['Ali Wong', 'Anthony Jeselnik', 'Bill Burr', 'Dave Chappelle',
             'Jim Jefferies', 'Joe Rogan', 'John Mulaney', 'Louis C.K.', 'Mike Bi

data = {}

for i, c in enumerate(comedians):
    print(c, transcripts[i])
    data[c] = transcripts[i]
```

```
Ali Wong ['Ladies and gentlemen, please welcome to the stage: Ali Wong! Hi. Hell
Anthony Jeselnik ['Thank you. Thank you. Thank you, San Francisco. Thank you so
Bill Burr ['[cheers and applause] All right, thank you! Thank you very much! Tha
Dave Chappelle ['This is Dave. He tells dirty jokes for a living. That stare is
Jim Jefferies ['[Car horn honks] [Audience cheering] [Announcer] Ladies and gent
Joe Rogan ['[rock music playing]', '[audience cheering]', '[announcer]', 'Ladies
John Mulaney ['Armed with boyish charm and a sharp wit, the former “SNL” writer
Louis C.K. ['Intro\nFade the music out. Let’s roll. Hold there. Lights. Do the l
Mike Birbiglia ['Wow. Hey, thank you. Thanks. Thank you, guys. Hey, Seattle. Nic
Ricky Gervais ['Hello. Hello! How you doing? Great. Thank you. Wow. Calm down. S
```

```
# Check the data
data.keys()
```

```
dict_keys(['Ali Wong', 'Anthony Jeselnik', 'Bill Burr', 'Dave Chappelle', 'Jim J
```

```
# Check the data
data['John Mulaney']
```

```
['Armed with boyish charm and a sharp wit, the former “SNL” writer John Mulaney
 'All right, Petunia. Wish me luck out there. You will die on August 7th, 2037.
 'You’re very friendly here in Chicago. I mean, we’re all violent here, but you’
 "It’s been a while since I’ve been home to Chicago. I got married since then. T
 'I talked to a lot of people before I got engaged, you know. And I heard this e
 'My wife is Jewish. She’s a New York Jew. I did it! Now, I was raised Catholic.
 "I grew up Catholic. I don’t go to church anymore. But I went on Christmas Eve
 'My wife and I don’t have any children, we have a dog. We have a little puppy n
 'I have a wife and a dog, and we just bought a house. We have a new house. It w
 'I loved our real estate agent. It was so fun to hang out with her. It was like
 'I didn’t mean to make it sound like we don’t want children. We don’t, but I di
 'It’s fun to be married. I’ve never been supervised before. I’m supervised. She
 'And there weren’t special things for kids the way there are now. Like, we woul
 'Kids have it very good now. My friend’s a teacher. She told me that, uh… the p
 'No, my parents loved us. It’s just, like, they were the cops, you know? And we
 "My dad loved us. He just didn’t care about our general happiness or self-esteem
 'The weirdest thing when I was a kid was how much they scared us about smoking
 'What are you, on your phone? Hey, V-neck. Hey! — What’s your name? — Sam. Sam?
 'I was an office temp for a while. I really miss that. I loved being a temp, be
```

```
 "I temped at a little web company on 25th Street in New York City. It was a sma
 'That’s the wonderful thing about crazy people, you know? Is that they just hav
 'I wanna tell you one more story before I get out of here, about the night I me
 'That is not the Bill Clinton that we all signed up for 20 years ago. Our Bill
 'And I got to meet Bill Clinton because my parents had gone to the same college
 'My mom loved Bill Clinton, ’cause Bill Clinton was always a really charismatic
 'Now, my dad, on the other hand, hated Bill Clinton, because my parents were da
 'So, one day, this invitation arrives for a fundraiser where you could meet Bil
 'I learned to play his campaign song on the piano. It was “Don’t Stop” by Fleetw
 'Anyway, so it’s that ballroom. So, we walk into that ballroom. It was packed w
 'We land at Bill Clinton’s feet. Bill Clinton turns, looks at my mom and says,
 'And I got home that night… I got home that night, and my dad was still awake,
 'Let’s flash forward five years to 1997. It is now 1997. I am a sophomore in hi
 'Good night, Chicago.',
 'Your email address will not be published. Required fields are marked *',
 'Name*',
 'E-mail*',
 'Website',
 ' \n\n',
 'Stand-up special by Jim Gaffigan.',
 'Louis C.K. being Louis C.K. without restraint.',
 'Comedian Drew Michael is taking the stage and is holding nothing back in his f
 'Follows Drew Michael and his issues with relationships, social media, and come
```

Cleaning the data

```
# Current format of the dictionary: "comedian": ["text1", "text2", ..] (key: comedi
# Let's change it to: "comedian": ["large text"] (key: comedian, value: list of all
data_combined = {comedian: ["".join(list_of_text)] for (comedian, list_of_text) in
data_combined
```

```
{'Ali Wong': ["Ladies and gentlemen, please welcome to the stage: Ali Wong! Hi.
 'Anthony Jeselnik': ["Thank you. Thank you. Thank you, San Francisco. Thank you
 'Bill Burr': ['[cheers and applause] All right, thank you! Thank you very much!
 'Dave Chappelle': ['This is Dave. He tells dirty jokes for a living. That stare
 'Jim Jefferies': ["[Car horn honks] [Audience cheering] [Announcer] Ladies and
 'Joe Rogan': ['[rock music playing][audience cheering][announcer]Ladies and gen
 'John Mulaney': ["Armed with boyish charm and a sharp wit, the former “SNL” wri
 'Louis C.K.': ['Intro\nFade the music out. Let’s roll. Hold there. Lights. Do t
 'Mike Birbiglia': ['Wow. Hey, thank you. Thanks. Thank you, guys. Hey, Seattle.
 'Ricky Gervais': ['Hello. Hello! How you doing? Great. Thank you. Wow. Calm dow
```

```
# Let's transform the data into a pandas dataframe
pd.set_option('max_colwidth',180)
data_df = pd.DataFrame.from_dict(data_combined).transpose()
data_df.columns = ['transcript']
data_df = data_df.sort_index()
data_df
```

	transcript
Ali Wong	Ladies and gentlemen, please welcome to the stage: Ali Wong! Hi. Hello! Welcome! Thank you! Thank you for coming. Hello! Hello. We are gonna have to get this shit over with, 'c...
Anthony Jeselnik	Thank you. Thank you. Thank you, San Francisco. Thank you so much. So good to be here. People were surprised when I told 'em I was gonna tape my special in San Francisco. Said,...
Bill Burr	[cheers and applause] All right, thank you! Thank you very much! Thank you. Thank you. Thank you. How are you? What's going on? Thank you. It's a pleasure to be here in the gre...
Dave Chappelle	This is Dave. He tells dirty jokes for a living. That stare is where most of his hard work happens. It signifies a profound train of thought, the alchemist's fire that transfor...
Jim Jefferies	[Car horn honks] [Audience cheering] [Announcer] Ladies and gentlemen, please welcome to the stage Mr. Jim Jefferies! [Upbeat music playing] Hello! Sit down, sit down, sit down...
Joe Rogan	[rock music playing][audience cheering][announcer]Ladies and gentlemen, welcome Joe Rogan.[audience cheering and applauding]What the f*ck is going on, San Francisco? Thanks for...
John	Armed with boyish charm and a sharp wit, the former "SNL" writer John Mulaney offers sly

```
# Let's take a closer look at John Mulaney's transcript
data_df.transcript.loc['John Mulaney']
```

```
'Armed with boyish charm and a sharp wit, the former "SNL" writer John Mulaney o
ffers sly takes on marriage, his beef with babies and the time he met Bill Clint
onAll right, Petunia. Wish me luck out there. You will die on August 7th, 2037.
That's pretty good. All right. Hello. Hello, Chicago. Nice to see you again. Tha
nk you. That was very nice. Thank you. Look, now, you're a wonderful crowd, but
I need you to keep your energy up the entire show, okay? Because… No, no, no. Th
ank you. Some crowds… some crowds, they have big energy in the beginning and the
n they run out of places to go. So… I don't judge those crowds, by the way, oka
```

```
# Let's clean up the text
def clean_text(text):
    '''
    Make text lowercase, remove text in square brackets, remove punctuation and rem
    Get rid of additional punctuation and non-sensical text.
    '''
    text = text.lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\w*\d\w*', '', text)
    text = re.sub('[''""…]', '', text)
    text = re.sub('\n', '', text)
    return text

cleaning = lambda x: clean_text(x)
```

```
# Let's take a look at the updated text
data_clean = pd.DataFrame(data_df.transcript.apply(cleaning))
data_clean
```

	transcript
Ali Wong	ladies and gentlemen please welcome to the stage ali wong hi hello welcome thank you thank you for coming hello hello we are gonna have to get this shit over with cause i have ...
Anthony Jeselnik	thank you thank you thank you san francisco thank you so much so good to be here people were surprised when i told em i was gonna tape my special in san francisco said why woul...
Bill Burr	all right thank you thank you very much thank you thank you thank you how are you whats going on thank you its a pleasure to be here in the greater atlanta georgia area this o...
Dave Chappelle	this is dave he tells dirty jokes for a living that stare is where most of his hard work happens it signifies a profound train of thought the alchemists fire that transforms fe...
Jim Jefferies	ladies and gentlemen please welcome to the stage mr jim jefferies hello sit down sit down sit down sit down thank you boston i appreciate that uh thats very swee...
Joe Rogan	ladies and gentlemen welcome joe roganwhat the fck is going on san francisco thanks for coming i appreciate it god damn put your phone down fckface i see you btch put your phon...
John Mulaney	armed with boyish charm and a sharp wit the former snl writer john mulaney offers sly takes on marriage his beef with babies and the time he met bill clintonall right petunia w...
Louis	introfade the music out lets roll hold there lights do the lights thank you thank you very much i

Organizing the data

Document-term matrix

```
# We are going to create a document-term matrix using CountVectorizer, and exclude
from sklearn.feature_extraction.text import CountVectorizer

cv = CountVectorizer(stop_words='english')
data_cv = cv.fit_transform(data_clean.transcript)
data_dtm = pd.DataFrame(data_cv.toarray(), columns=cv.get_feature_names())
data_dtm.index = data_clean.index
data_dtm
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWa
  warnings.warn(msg, category=FutureWarning)
```

	aaaaah	aaah	aah	abc	abcs	ability	abject	able	ablebodied	abortio
Ali Wong	0	0	0	1	0	0	0	2	0	
Anthony Jeselnik	0	0	0	0	0	0	0	0	0	
Bill Burr	1	0	0	0	1	0	0	1	0	
Dave Chappelle	0	1	0	0	0	0	0	0	0	
Jim Jefferies	0	0	0	0	0	0	0	1	2	
Joe Rogan	0	0	0	0	0	0	0	2	0	
John Mulaney	0	0	0	0	0	0	0	3	0	
Louis C.K.	0	0	3	0	0	0	0	1	0	

Exploratory Data Analysis

Most common words

10 rows × 6791 columns

```
# Read in the document-term matrix
import copy

data = copy.deepcopy(data_dtm)
data = data.transpose()
data.head()
```

	Ali Wong	Anthony Jeselnik	Bill Burr	Dave Chappelle	Jim Jefferies	Joe Rogan	John Mulaney	Louis C.K.	Mi Birbigl
aaaaah	0	0	1	0	0	0	0	0	
aaah	0	0	0	1	0	0	0	0	
aah	0	0	0	0	0	0	0	3	
abc	1	0	0	0	0	0	0	0	
abcs	0	0	1	0	0	0	0	0	

```
# Find the top 10 words said by each comedian
top_dict = {}
for c in data.columns:
```

```
    top = data[c].sort_values(ascending=False).head(30)
    top_dict[c]= list(zip(top.index, top.values))

# Print the top 15 words said by each comedian
for comedian, top_words in top_dict.items():
    print(comedian)
    print(', '.join([word for word, count in top_words]))
    print('---')
```

```
Ali Wong
like, im, just, know, dont, thats, shit, youre, gonna, ok, lot, wanna, oh, gotta
---
Anthony Jeselnik
im, like, know, dont, joke, got, said, thats, anthony, day, say, just, guys, peo
---
Bill Burr
like, just, right, im, know, dont, gonna, got, fucking, yeah, shit, youre, thats
---
Dave Chappelle
like, know, said, just, im, shit, people, didnt, dont, ahah, time, thats, fuck,
---
Jim Jefferies
like, im, dont, right, fucking, went, know, just, youre, people, thats, day, oh,
---
Joe Rogan
like, people, just, dont, im, fcking, fck, thats, gonna, theyre, know, youre, th
---
John Mulaney
like, know, just, dont, said, im, clinton, thats, right, youre, little, hey, tim
---
Louis C.K.
like, just, know, dont, thats, youre, im, people, life, thing, hes, gonna, there
---
Mike Birbiglia
like, im, know, said, just, dont, think, thats, says, cause, right, jenny, goes,
---
Ricky Gervais
right, like, just, im, dont, know, said, yeah, fucking, got, say, youre, went, i
---
```

```
# Look at the most common top words --> add them to the stop word list
from collections import Counter

# Let's first pull out the top 30 words for each comedian
words = []
for comedian in data.columns:
    top = [word for (word, count) in top_dict[comedian]]
    for t in top:
        words.append(t)

words
```

```
 'know',
 'said',
 'just',
```

```
 'dont',
 'think',
 'thats',
 'says',
 'cause',
 'right',
 'jenny',
 'goes',
 'really',
 'id',
 'point',
 'mean',
 'youre',
 'gonna',
 'yeah',
 'got',
 'kind',
 'people',
 'uh',
 'say',
 'didnt',
 'going',
 'want',
 'time',
 'oh',
 'right',
 'like',
 'just',
 'im',
 'dont',
 'know',
 'said',
 'yeah',
 'fucking',
 'got',
 'say',
 'youre',
 'went',
 'id',
 'thats',
 'people',
 'didnt',
 'little',
 'joke',
 'theyre',
 'hes',
 'ive',
 'going',
 'thing',
 'years',
 'day',
 'theres',
 'saying',
 'big',
 'hed']
```

```
# If more than half of the comedians have it as a top word, exclude it from the lis
add_stop_words = [word for word, count in Counter(words).most_common() if count > 6
add_stop_words
```

```
['like',
 'im',
 'just',
 'know',
 'dont',
 'thats',
 'right',
 'people',
 'youre',
 'got',
 'gonna',
 'think']
```

```
# Let's update our document-term matrix with the new list of stop words
from sklearn.feature_extraction import text
from sklearn.feature_extraction.text import CountVectorizer

# Read in cleaned data
# data_clean = pd.read_pickle('data_clean.pkl')

# Add new stop words
stop_words = text.ENGLISH_STOP_WORDS.union(add_stop_words)

# # Recreate document-term matrix
# cv = CountVectorizer(stop_words=stop_words)
# data_cv = cv.fit_transform(data_clean.transcript)
# data_stop = pd.DataFrame(data_cv.toarray(), columns=cv.get_feature_names())
# data_stop.index = data_clean.index

# # Pickle it for later use
# import pickle
# pickle.dump(cv, open("cv_stop.pkl", "wb"))
# data_stop.to_pickle("dtm_stop.pkl")
```

```
# Let's make some word clouds!
# Terminal / Anaconda Prompt: conda install -c conda-forge wordcloud
from wordcloud import WordCloud

wc = WordCloud(stopwords=stop_words, background_color="white", colormap="Dark2",
               max_font_size=150, random_state=42)
```

```
# Reset the output dimensions
import matplotlib.pyplot as plt

```

```
plt.rcParams['figure.figsize'] = [20, 10]

# full_names = ['Ali Wong', 'Anthony Jeselnik', 'Bill Burr', 'Dave Chappelle', 'Jir

# Create subplots for each comedian
for index, comedian in enumerate(data.columns):
    wc.generate(data_clean.transcript[comedian])

    plt.subplot(2, 5, index+1)
    plt.imshow(wc, interpolation="bilinear")
    plt.axis("off")
    plt.title(comedians[index])

plt.show()
```

Sentiment Analysis

Sentiment of Routine

```
# Let's get our clean data
data = copy.deepcopy(data_clean)
data
```

	transcript
Ali Wong	ladies and gentlemen please welcome to the stage ali wong hi hello welcome thank you thank you for coming hello hello we are gonna have to get this shit over with cause i have ...
Anthony Jeselnik	thank you thank you thank you san francisco thank you so much so good to be here people were surprised when i told em i was gonna tape my special in san francisco said why woul...
Bill Burr	all right thank you thank you very much thank you thank you thank you how are you whats going on thank you its a pleasure to be here in the greater atlanta georgia area this o...
Dave Chappelle	this is dave he tells dirty jokes for a living that stare is where most of his hard work happens it signifies a profound train of thought the alchemists fire that transforms fe...
Jim Jefferies	ladies and gentlemen please welcome to the stage mr jim jefferies hello sit down sit down sit down sit down sit down thank you boston i appreciate that uh thats very swee...
Joe Rogan	ladies and gentlemen welcome joe roganwhat the fck is going on san francisco thanks for coming i appreciate it god damn put your phone down fckface i see you btch put your phon...
John Mulaney	armed with boyish charm and a sharp wit the former snl writer john mulaney offers sly takes on marriage his beef with babies and the time he met bill clintonall right petunia w...

```
# Create quick lambda functions to find the polarity and subjectivity of each routi
# 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['full_name'] = comedians
data
```

	transcript	polarity	subjectivity	full_name
Ali Wong	ladies and gentlemen please welcome to the stage ali wong hi hello welcome thank you thank you for coming hello hello we are gonna have to get this shit over with cause i have ...	0.066919	0.479126	Ali Wong
Anthony Jeselnik	thank you thank you thank you san francisco thank you so much so good to be here people were surprised when i told em i was gonna tape my special in san francisco said why woul...	0.048833	0.560856	Anthony Jeselnik

```
# 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

```
# Show length of the script for each
for index, row in data.iterrows():
    print(row["full_name"], len(row["transcript"]))
```

```
Ali Wong 37234
Anthony Jeselnik 33134
Bill Burr 60485
Dave Chappelle 45718
Jim Jefferies 54104
Joe Rogan 50046
John Mulaney 46003
Louis C.K. 37139
Mike Birbiglia 55695
Ricky Gervais 53368
```

```
# Split each routine into 10 parts
import numpy as np
import math

def split_text(text, n=200):
    '''
    Takes in a string of text and splits into n equal parts, with a default of 200
    With a total length of script from 30,000 to 60,000 characters, we are looking
    '''

    # Calculate length of text, the size of each chunk of text and the starting poi
    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
```

	transcript	polarity	subjectivity	full_name
Ali Wong	ladies and gentlemen please welcome to the stage ali wong hi hello welcome thank you thank you for coming hello hello we are gonna have to get this shit over with cause i have ...	0.066919	0.479126	Ali Wong
Anthony Jeselnik	thank you thank you thank you san francisco thank you so much so good to be here people were surprised when i told em i was gonna tape my special in san francisco said why woul...	0.048833	0.560856	Anthony Jeselnik
Bill Burr	all right thank you thank you very much thank you thank you thank you how are you whats going on thank you its a pleasure to be here in the greater atlanta georgia area this o...	0.008194	0.543606	Bill Burr
Dave Chappelle	this is dave he tells dirty jokes for a living that stare is where most of his hard work happens it signifies a profound train of thought the alchemists fire that transforms fe...	-0.003324	0.513580	Dave Chappelle
Jim Jefferies	ladies and gentlemen please welcome to the stage mr jim jefferies hello sit down sit down sit down sit down sit down thank you boston i appreciate that uh thats very swee...	0.038305	0.537510	Jim Jefferies
Joe Rogan	ladies and gentlemen welcome joe roganwhat the fck is going on san francisco thanks for coming i appreciate it god damn put your phone down fckface i see you btch put your	0.077798	0.535533	Joe Rogan

```
1 # Let's create a list to hold all of the pieces of text
2 list_pieces = []
3 for t in data.transcript:
4     split = split_text(t)
5     list_pieces += split
6
7 list_pieces
```

```
 at never that moment but by golly if its not fun i find it to look to all the
' during memorial day after the vietnam vets before the first gulf war guys we
'ard tongue out and the like that i dont know why but i enjoy i know its a lie
'rgasm anything could happen in this crazy world but this is what redeems us as
'i go from being an animal to the sweetest guy on earth im like you fucking slu
'oure a wonderful mother to our child   see this is what kills me my son will o
'might be illegal tell me if this is illegal all right im in the shower my girl
'i saw his little face and i went hello hankie and then in the condensation i d
'i stepped out i stepped out of the bathroom and i went gday hankie and he slap
'ry at him because his whole life hes been lying on mats with things dangling o
'think like an american person now and im happy to do it i just ill tell you wh
'en you go i was flying sydney to melbourne when you fly domestically in austra
'ur bag on bag fucks off you dont speak to anyone then i go up to the gate bit
'ed to see that and i went i think you do and she went i dont why would i need
'veyor belt thing and im so good at the airports im already taking my shoes off
' mate what are you taking your shoes off for and i went  i dont know maybe the
```

```
'z mate thats a nice computer why are you showing it to everyone it might also l
'ans but it does make some type of senseall right oscar pistorius if you havent
' is a legless man from south africa known as the blade runner he ran in two oly
'last year he shot and killed the hottest girl on earth and thats when he becam
'u want whenever you fucking want  people have feelings you cunts now theres a
'african rugby player on valentines day right now i dont know if youve ever beer
'im going to reenact what i believe happened that day to do that i will now be
'what im trying to say is south africans are horrible people so shes coming out

's ive been through your phone you have been texting a rugby player and shes lil
't least he is a whole man not a threequarter man like you —  — i know  oh fuck
' go anywhere  i hate you thats where he keeps his legs all right then he put th
' little tiny locks on them but oscars one of the few men on earth that couldnt
'prison population now can i say this look ive never raped a man i hate that i l
'le and making people think youre tough and all that like you want to fucking ra
' what i like about that joke so often when you tell a joke the rapist is the vi
'on you cunts so calm down   i always i always find that weird when someone like
'od you came along to pick up my spirits  love you all right final story now i w
'ited this time theyll be like  we dont like him hes no good anyway so im in sou
's ticket so i didnt give a fuck and when i travel economy i try to dress up nic
'e that everyone else in business class doesnt want me there and theyre annoyed
'ter up to the business thing with the thing and i go hello and the lady goes
'ht one of the tickets that made it full and she went im sorry sir theres nothir
' engage with you anymore they can act like theyre the first adult never to hear
'south african telling me youve done nothing wrong anyway tensions rose the mana
'and he goes what do you want me to do make a new chair for you there are no mor
'ack up to business class so i thought theres nothing i can do so i walk off wit
'er my ticket and she goes  im sorry sir but this is for business class passengc
' here you fucking cunt  all right now you say cunt in any foreign country peopl
'manager up and then they go he goes is it the pale australian man and the guy w
'g like that and in walks in a group of americans about  of them you know the ty
'oing  oh this is great look at that thats a chair right there okay oh whats tha
'here some people have no class and i put my head around the pylon and i went yo
't people promoted up to business class the penny dropped none of us are gettinc
'doing this dont speak to me that way who the hell do you think you are and all
'm everyone down and sometimes americans sometimes you can seem a little insinc
'at least this guys being nice and i said look mate dont worry about it its not
'hat neil diamond walked around the corner like a fucking superhero and i reacte
'd i went neil me and all these people weve been downgraded and neil went oh oh
'hes are thrown not by me i dont know if you have the internet but im not much
'sted from the neil diamond band and that meant that three seats opened up in bu
'meemailwebsite standup special by jim gaffiganlouis ck being louis ck without
...]
```

```
1 # The list has been broken down to 2000 transcripts
2 len(list_pieces)
```

```
2000
```

```
1 # Create list of comedian names
2
3 def create_big_list(list_of_names, n=200):
4     big_list = []
5     for name in list_of_names:
6         small_list = [name for i in range(n)]
```

```
        big_list += small_list
    return big_list

comedians = ['Ali Wong', 'Anthony Jeselnik', 'Bill Burr', 'Dave Chappelle', 'Jim Je
             'Joe Rogan', 'John Mulaney', 'Louis C.K.', 'Mike Birbiglia', 'Ricky Ge

comedians_list = create_big_list(comedians)
print(len(comedians_list))
print(comedians_list[:5])
```

```
2000
['Ali Wong', 'Ali Wong', 'Ali Wong', 'Ali Wong', 'Ali Wong']
```

```
# Calculate the polarity for each piece of text

NEUTRAL = "neutral"
POSITIVE = "positive"
NEGATIVE = "negative"

polarity_scores = []
sentiment_labels = []
sentiment_nums = []

for lp in list_pieces:
    polarity_score = TextBlob(lp).sentiment.polarity
    if -0.2 <= polarity_score <= 0.2:
        label = NEUTRAL
        num = 1
    elif polarity_score > 0.2:
        label = POSITIVE
        num = 2
    else:
        label = NEGATIVE
        num = 0

    polarity_scores.append(polarity_score)
    sentiment_labels.append(label)
    sentiment_nums.append(num)

# Take a peek at it
print(len(polarity_scores))
print(polarity_scores[:10])
print("----------")
print(len(sentiment_labels))
print(sentiment_labels[:10])
```

```
2000
[0.4666666666666673, 0.24500000000000002, -0.16666666666666669, 0.1551020408163
----------
2000
['positive', 'positive', 'neutral', 'neutral', 'positive', 'positive', 'positive
```

```
print(list_pieces[0])
print(list_pieces[1])
print(list_pieces[2])
print(list_pieces[3])
```

```
ladies and gentlemen please welcome to the stage ali wong hi hello welcome thank
like ten minutes but thank you everybody so much for comingum its a very excitin
te that uh i can tell that im getting older because now when i see an  girl my a
alous first and foremost of their metabolism because  girls they could just eat
```

```
# Put all of them together
new_data_df = pd.DataFrame(comedians_list, columns =['full_name'])
new_data_df["splitted_transcript"] = list_pieces
new_data_df["sentiment"] = sentiment_labels
new_data_df["y"] = sentiment_nums
new_data_df["score"] = polarity_scores
new_data_df
```

	full_name	splitted_transcript	sentiment	y	score
4	Ali Wong	that huge gap here with the light of potential just	positive	2	0.271429
	Ricky	british bobby there and he stops the car bobs at the			

```
# Prepare data for classification
X = new_data_df["splitted_transcript"].values
Y = new_data_df["y"].values
```

```
# Split data to train and test dataset
from sklearn.model_selection import train_test_split

X_train, X_test, Y_train, Y_test= train_test_split(X,Y, test_size=0.3)
Y_train = Y_train.reshape(-1, 1)
Y_test = Y_test.reshape(-1, 1)
```

```
from sklearn.feature_extraction.text import CountVectorizer

vec = CountVectorizer()
vec.fit(X_train)
x_train=vec.transform(X_train)
x_test=vec.transform(X_test)
```

```
Y_train.shape
```

```
(1400, 1)
```

```
x_train
```

```
<1400x6296 sparse matrix of type '<class 'numpy.int64'>'
        with 49805 stored elements in Compressed Sparse Row format>
```

```
# first neural network with keras tutorial
from keras.models import Sequential
from keras.layers import Dense
from sklearn import datasets
import matplotlib.pyplot as plt
import numpy as np
```

```
# Train and test with Neural Network
from keras.models import Sequential
from keras.layers import Dense

model = Sequential()
model.add(Dense(100, input_dim=x_train.shape[1], activation='relu'))
model.add(Dense(32, activation='relu'))
model.add(Dense(3, activation='sigmoid'))
model.summary()
```

```
Model: "sequential_2"
```

```
 Layer (type)                 Output Shape              Param #
=================================================================
 dense_6 (Dense)              (None, 100)               629700

 dense_7 (Dense)              (None, 32)                3232

 dense_8 (Dense)              (None, 3)                 99

=================================================================
Total params: 633,031
Trainable params: 633,031
Non-trainable params: 0
_________________________________________________________________
```

```
1 model.compile(loss = 'sparse_categorical_crossentropy',
2               optimizer = 'adam',
3               metrics=['accuracy'])
```

```
1 model.fit(x_train, Y_train, epochs=10,verbose=True,batch_size=16)
```

```
Epoch 1/10
/usr/local/lib/python3.7/dist-packages/tensorflow/python/framework/indexed_slices
  "shape. This may consume a large amount of memory." % value)
88/88 [==============================] - 2s 15ms/step - loss: 0.8994 - accuracy:
Epoch 2/10
88/88 [==============================] - 1s 11ms/step - loss: 0.4734 - accuracy:
Epoch 3/10
88/88 [==============================] - 1s 14ms/step - loss: 0.1137 - accuracy:
Epoch 4/10
88/88 [==============================] - 1s 14ms/step - loss: 0.0199 - accuracy:
Epoch 5/10
88/88 [==============================] - 1s 10ms/step - loss: 0.0054 - accuracy:
Epoch 6/10
88/88 [==============================] - 1s 14ms/step - loss: 0.0027 - accuracy:
Epoch 7/10
88/88 [==============================] - 1s 13ms/step - loss: 0.0016 - accuracy:
Epoch 8/10
88/88 [==============================] - 1s 13ms/step - loss: 0.0011 - accuracy:
Epoch 9/10
88/88 [==============================] - 1s 13ms/step - loss: 8.2659e-04 - accura
Epoch 10/10
88/88 [==============================] - 1s 10ms/step - loss: 6.2602e-04 - accura
<keras.callbacks.History at 0x7f6dc8776090>
```

```
1 model.evaluate(x_train,Y_train)
```

```
44/44 [==============================] - 0s 2ms/step - loss: 5.2160e-04 - accurac
[0.0005215982091613114, 1.0]
```

```
1 model.evaluate(x_test,Y_test)
```

```
19/19 [==============================] - 0s 3ms/step - loss: 1.4864 - accuracy: 
[1.4863659143447876, 0.6416666507720947]
```

```
# Train and test with Logistics Regression
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression(solver='lbfgs', max_iter=100)
lr.fit(x_train, Y_train.ravel())
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:818: Co
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressio
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG,
LogisticRegression()
```

```
lr.score(x_train, Y_train)
```

```
1.0
```

```
lr.score(x_test, Y_test)
```

```
0.62
```

TF-IDF Vectorizer

```
from sklearn.feature_extraction.text import TfidfVectorizer
vec = TfidfVectorizer()
vec.fit(X_train)
```

```
TfidfVectorizer()
```

```
x_train=vec.transform(X_train)
x_test=vec.transform(X_test)
x_train
```

```
<1400x6296 sparse matrix of type '<class 'numpy.float64'>'
        with 49805 stored elements in Compressed Sparse Row format>
```

```
# Train and test with Logistics Regression
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression(solver='lbfgs', max_iter=100)
lr.fit(x_train, Y_train.ravel())
```

```
LogisticRegression()
```

```
1 lr.score(x_train, Y_train)
```

```
0.7828571428571428
```

```
1 lr.score(x_test, Y_test)
```

```
0.615
```

✓ 22s completed at 10:10 AM