WHO AM I READING ?

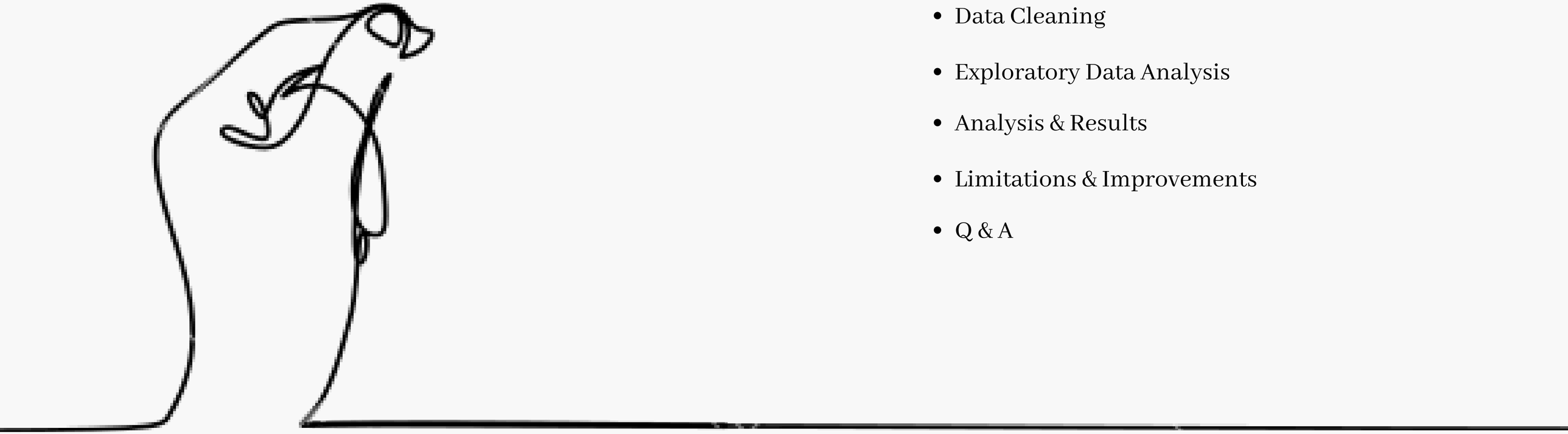
Analysis of blog authorship using
Natural Language Processing

Yunke Xiong

Ironhack

March 18th, 2023

AGENDA



- Case Overview
- Dataset Overview
- Data Cleaning
- Exploratory Data Analysis
- Analysis & Results
- Limitations & Improvements
- Q & A

CASE OVERVIEW

Have you ever wandered ...
what do people feel about a certain topic ?
who is behind the screen when you read a blog or a comment ?

Dataset:

- Blog Authorship Corpus

Analysis:

- Sentiment Analysis
- Author Profiling
- Topic Analysis

Results :

- Model Comparison
- Blogger Lifecycle

DATASET OVERVIEW

Dataset : Blog Authorship Corpus

Datasource : Kaggle

Content : 19,320 bloggers, 681,288 posts gathered from blogger.com in August 2004

Columns:

- id
- gender
- age
- topic (occupation)
- sign
- date
- text

DATASET CLEANING

basic cleaning:

- dropna
- remove rows without job information
- clean up date format (multi languages
- remove rows with wrong date

transformation:

- assign unique blog_id
- add word_count

```python
def clean_and_transform(data):
    # remove rows without topic and date and reset index
    data = data.loc[data['topic']!='indUnk',:]
    data = data.dropna()
    data = data.reset_index(drop = True)

    # create column for word count
    data.loc[:,'word_count'] = data.loc[:,'text'].apply(lambda x: len(x.split()))

    # create unique id for each blog
    data.loc[:, 'blog_id'] = data.index.astype(str)
    data.loc[:, 'blog_id'] = data.blog_id.apply(lambda x : '1'+x.zfill(6))

    # clean up the date in multiple language
    for ind, row in data.iterrows():
        try:
            date_raw = row['date']
            clean_date = dateparser.parse(date_raw)
            data.loc[ind,'date_clean'] = clean_date
        except Exception as e:
            print(ind, e)

    # after cleaning there're extra rows for data after 2004, remove those
    data = data.loc[data['date_clean']<'2005-01-01',:]

    # rename and arrange the columns
    data_clean = data.rename(columns = {'id':'author_id', 'date':'date_raw', 'date_clean': 'date','topic':'occupation'})
    data_clean = data_clean.loc[:,['author_id', 'gender', 'age','occupation', 'sign', 'blog_id','date','text', 'word_count']]

    return data_clean
```

DATASET CLEANING

text cleaning :

- library: spacy
- tokenize
- lemmatize
- lower case
- remove symbols, extra space, digits etc.
- remove stop words and words with 1 letter
- remove urllinks
- join back to clean text

```
nlp = spacy.load("en_core_web_sm")
stop_words = nlp.Defaults.stop_words
punctuations = string.punctuation
to_remove = ['urllink', 'jpg', 'nbsp', 'http', 'www']
```

```
def clean_text(sentence):
    import re
    # create token object
    doc = nlp(sentence)

    mytokens = [word.lemma_.lower().strip() for word in doc] # lemmatize token and convert into lower case
    mytokens = [re.sub(r'\W+',' ',token) for token in mytokens] # replace everything non-alpahnumeric by ' '
    mytokens = [re.sub(r'\s+',' ',token) for token in mytokens] # replace one or more whitespaces by  ' '
    mytokens = [re.sub(r'\d+',' ',token) for token in mytokens] # replace one or more digits by ' '

    # remove stopwords, punctuation and empty string
    mytokens = [word.strip() for word in mytokens if word not in stop_words
                and word not in punctuations and word not in to_remove and len(word) > 1]
    mytokens = [word for word in mytokens if word]
    clean_text = " ".join(mytokens)
    return clean_text
```

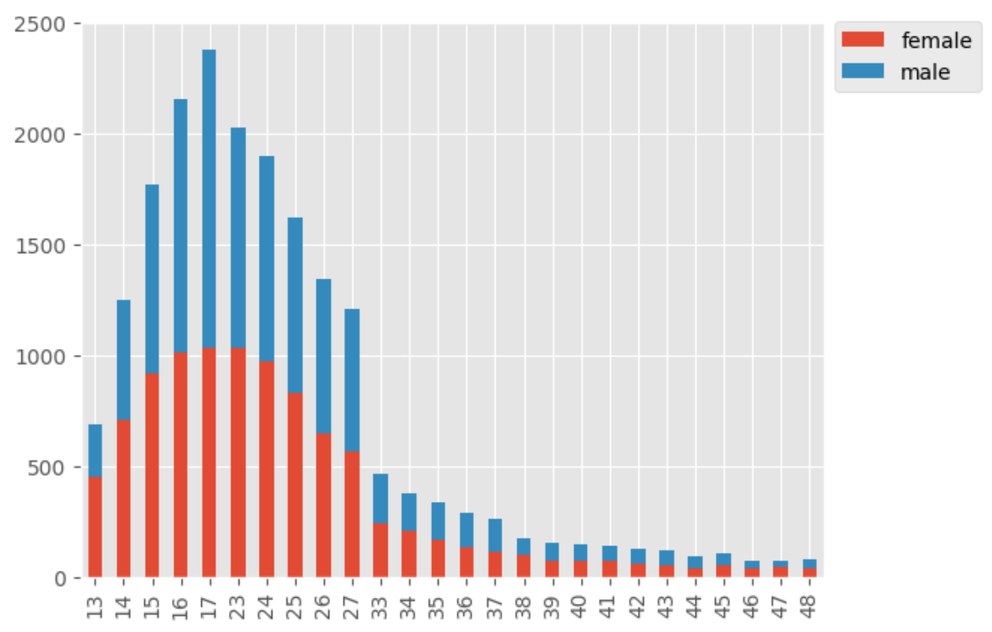
EXPLORATORY DATA ANALYSIS

Gender Composition :

male: 51%

female: 49%

Age Distribution of Bloggers:

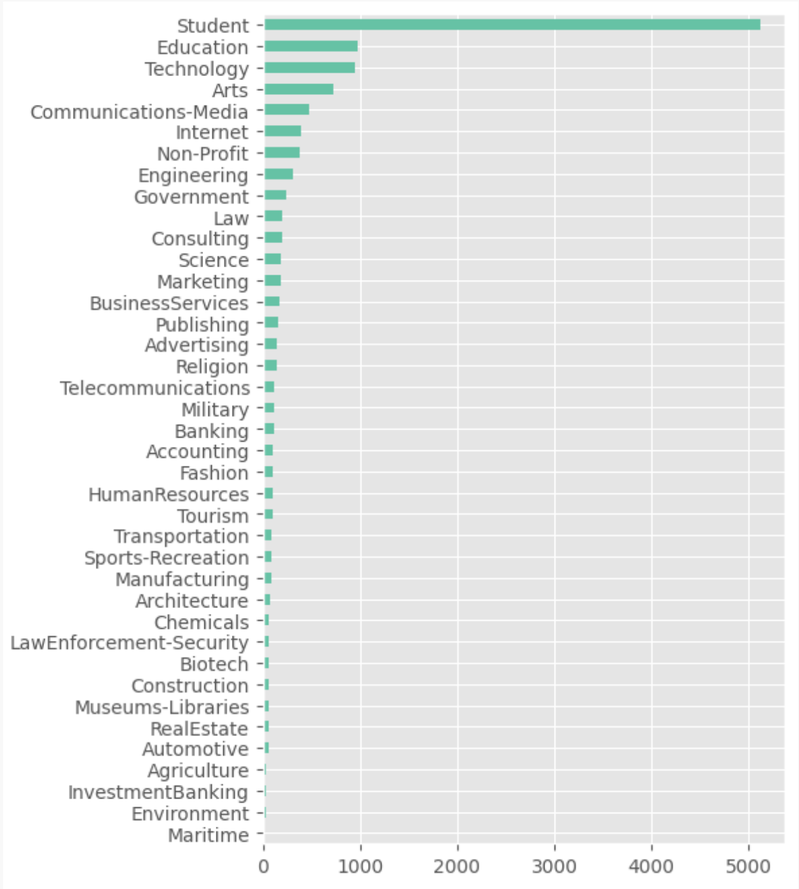


Job Distribution :

indUnk: 37%

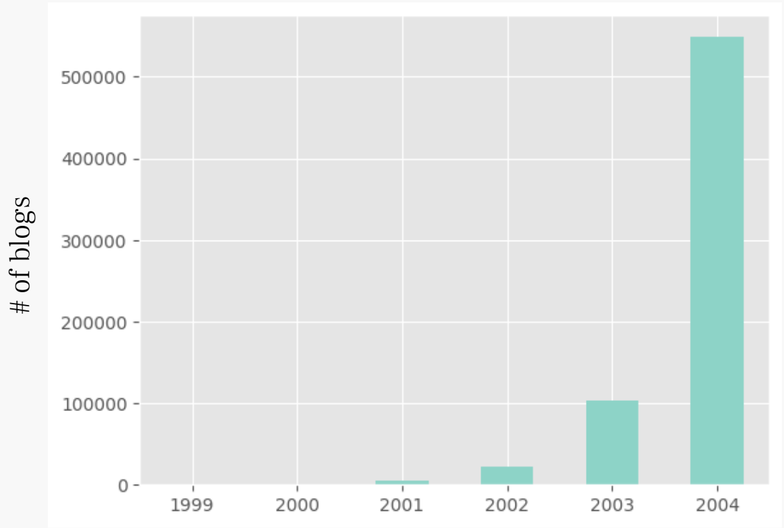
student: 23%

others: 41%

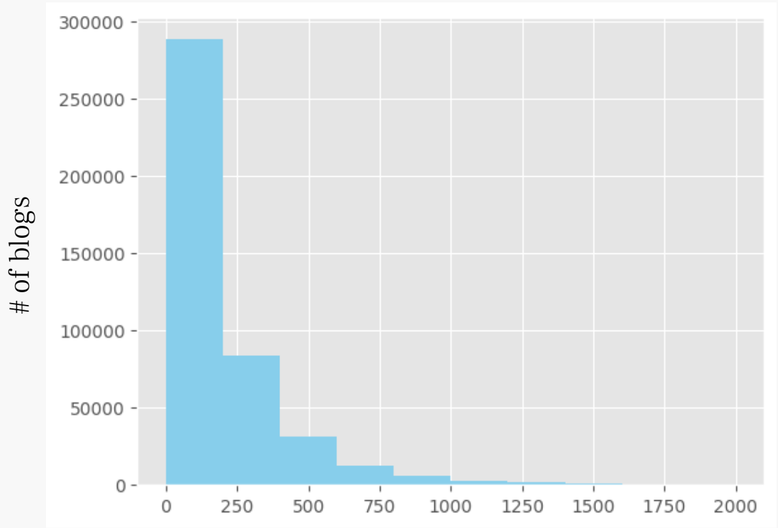


EXPLORATORY DATA ANALYSIS

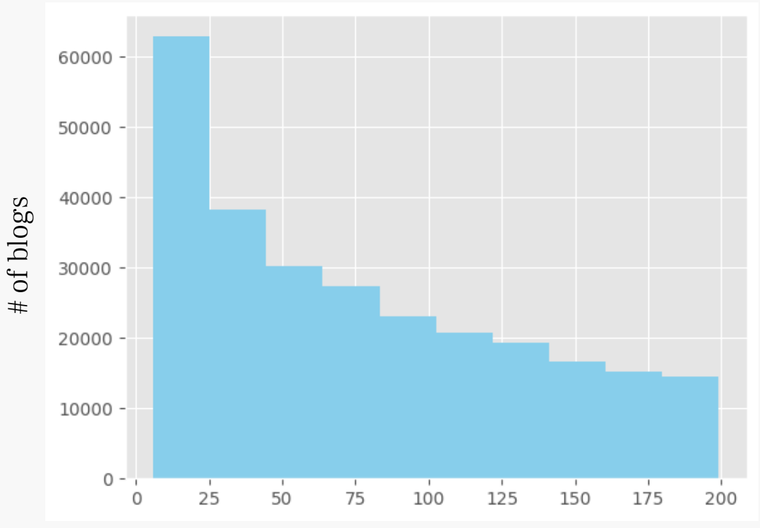
Year in which blogs posted



Word count per blog



Word count per blog - Zoom in



ANALYSIS & RESULTS

Sentiment Analysis

Methods: Sentiment Analysis

Models : Vader vs Roberta

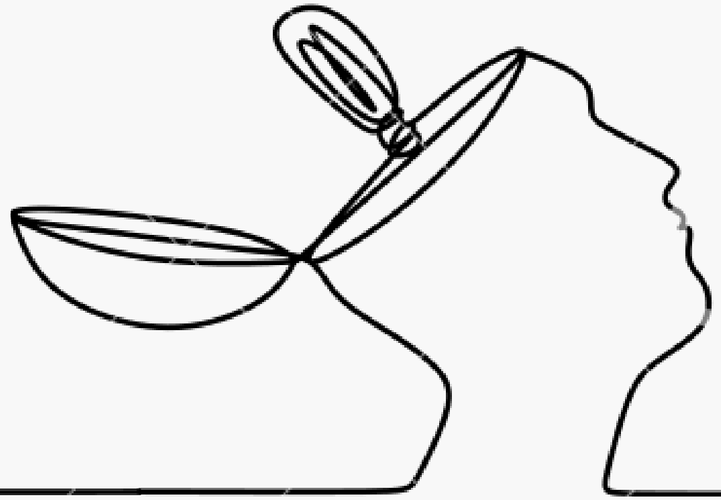
Author Profiling

Methods: Text Classification

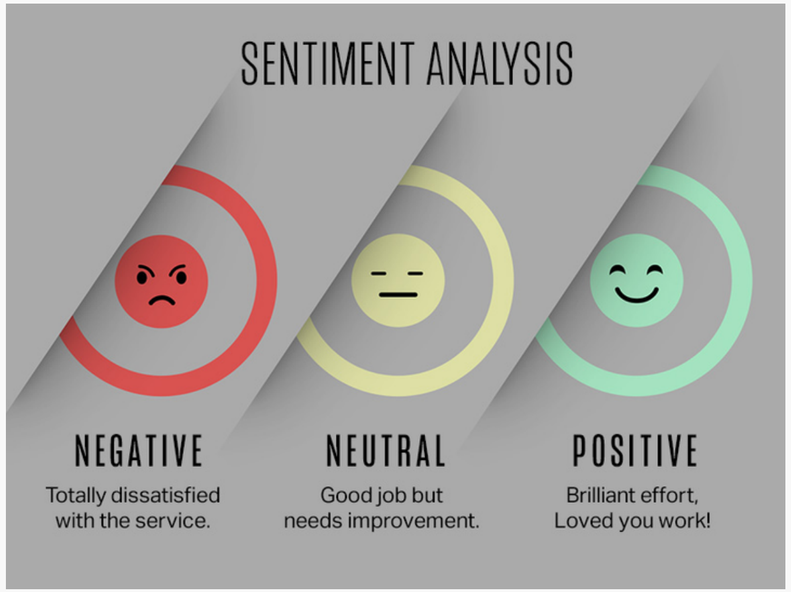
Models : Naive Bayes vs Logistics Resgression

Topic Analysis

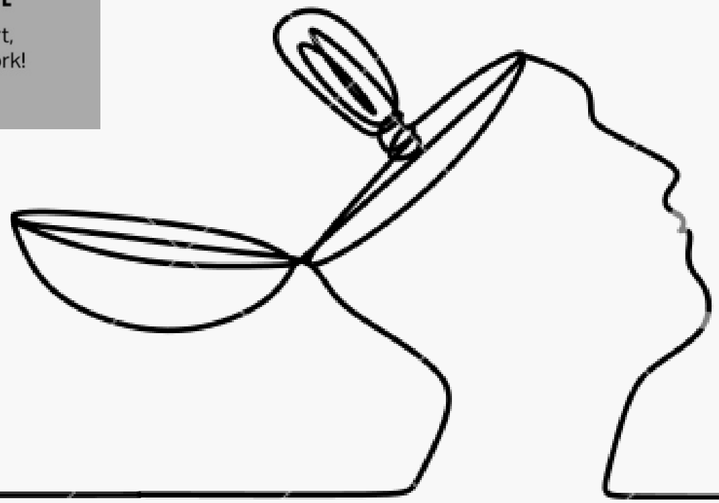
Methods: Word Frequency



SENTIMENT ANALYSIS



- data preparation
- showcase some examples
- model comparison
- what can we do with the result ?



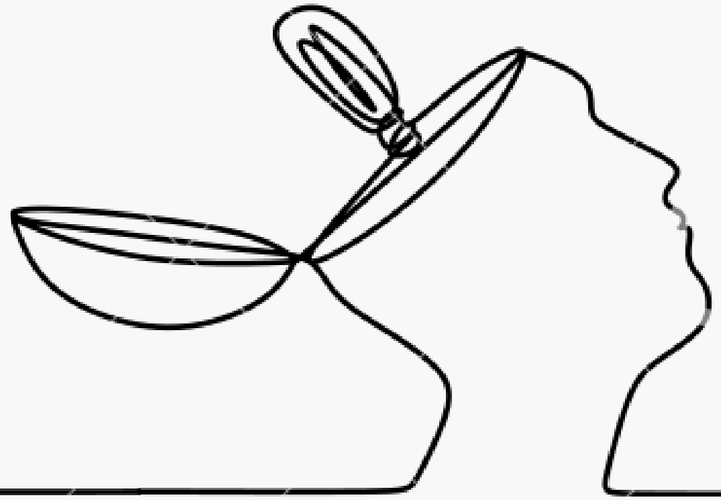
SENTIMENT ANALYSIS

20000 blogs used due to computing power constraint for Roberta (it takes 130 times longer than Vader !)

blogs with 5-200 words

```
data_raw = pd.read_csv('cleaned_text_dump.csv')
data_raw = data_raw[(data_raw['word_count']>5) & (data_raw['word_count']<200)]
data_raw = data_raw.dropna()
data_raw.loc[:,'gender'] = [1 if x =='female' else 0 for x in data_raw['gender']]
data_raw.loc[:,'age_coded'] = np.where(data_raw['age']<20,0,
                                       np.where(data_raw['age']<30,1,
                                       np.where(data_raw['age']<40,2,3)))
data_raw.reset_index(drop = True, inplace=True)
data_ml_sm = data_raw[:20000]
```

* original text used rather than cleaned



SENTIMENT ANALYSIS

VADER

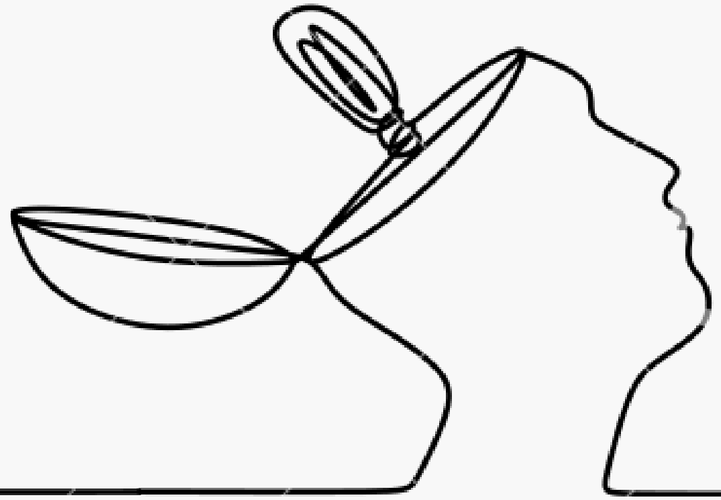
VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media, and works well on texts from other domains

library: nltk

ROBERTA

The RoBERTa model was proposed in RoBERTa: A Robustly Optimized BERT Pretraining Approach. It is based on Google's BERT model released in 2018.

library: transformers

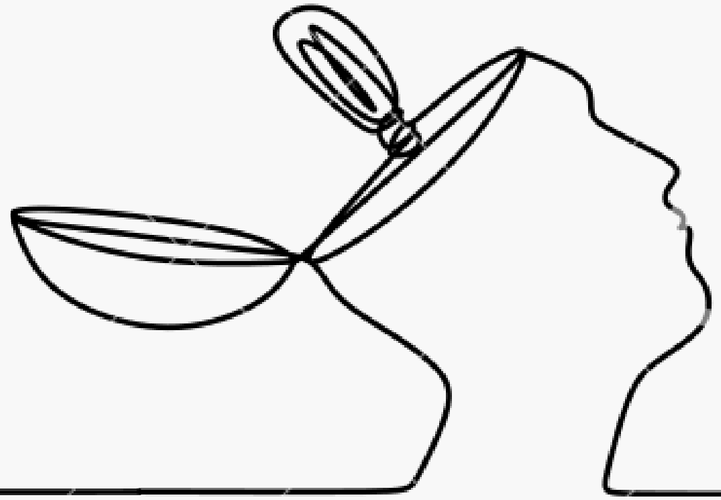


SENTIMENT ANALYSIS

blog 1:

..... **top five bad things** that have happened to me in the last week..... (not in order of importance) 1. i **sliced open my thumb** on the severed edge of the top of a cream of celery soup can. lots of blood. 2. i haven't had time to schedule a haircut. 3. i **didn't finish reading two papers** by my friday class last week. 4. i forgot to bring some stuff into school for my wife earlier today. 5. **something's wrong with my telephone** at home. What!? Is it my fault that I've got a good life? Did you *read* this blog last semester? Last semester **it sucked** to be me. I refuse to feel guilty for feeling ok about the world. (i'm sure i just **jinxed** everything though.) peace~

vader: {'neg': 0.145, 'neu': 0.767, 'pos': 0.088, 'compound': **-0.8788**}

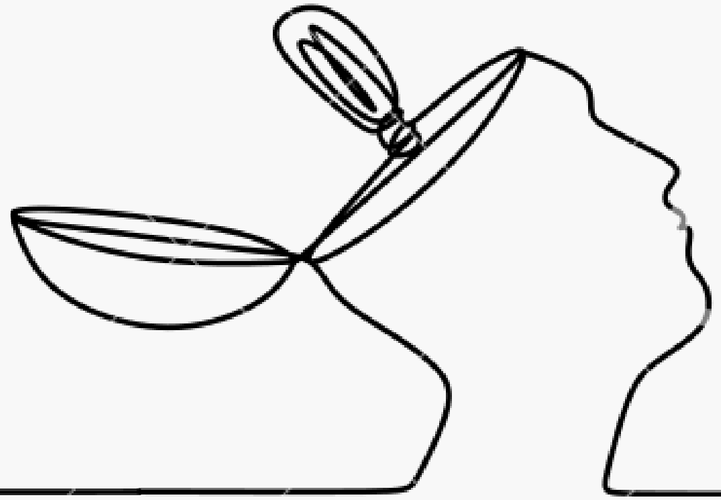


SENTIMENT ANALYSIS

blog 2:

Damn it's hump day. I was listening to the raadio today and they were talking about this event they're having comming up called Kegs and Eggs. It's a party from 8am-noon. **It sounds like a lot of fun**, but it's on a friday. It seems kinda **messed up** to take a whole day off just for a four hour party. And it seems like the think that you may stumble home from not walk. But Carbon Leaf is playing. I'll think about it, it's a month from friday (2 days prior to Jender's Birthday). On another note **I just ran out of money** (I mean until friday). **I kinda flipped out yesterday** about it. Then I realized I have another check before next rent. It's just with these alimony payments now, I have to rebudget. Yeah I know that's a lie, but I thought it was better than I spent $300 on a blanket and a pillow. Sometimes I just have to give myself a knowing look and say: 'Where's your head at?'

vader: {'neg': 0.027, 'neu': 0.847, 'pos': 0.127, 'compound': **0.9422**}

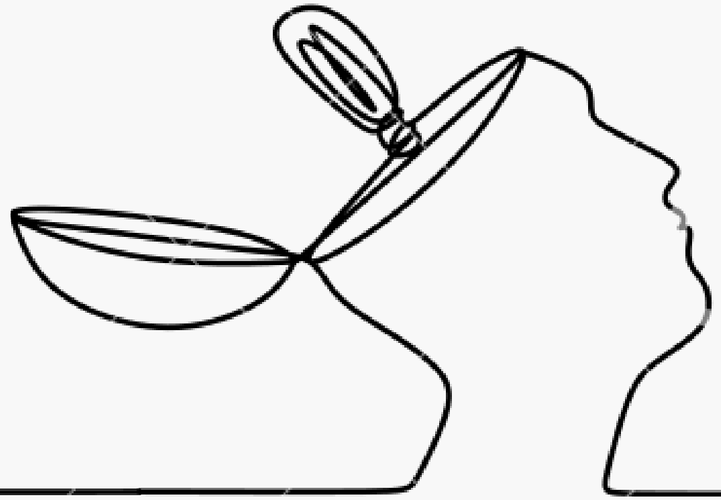


SENTIMENT ANALYSIS

blog 3:

Today was Ryan's sister's birthday. I was there at 9:30am and the party didnt start til 11:30am. But anyways I was there with Ryan. We hung out until the party and Ryan ate all the cheesecake and checked out the freshmen girls. Yeah **it was fun. Lol. I love Ryan he makes me laugh**! We had balloon fights...**I got into a fight** with Kyle and **that wasnt fun**. Well yeah **it was fun and I LOVE RYAN!** Today I went to work at Cazones.... **It was pretty fun** I met a new friend named Brandon. He thinks Iam beautiful. Lol. He told me when we were making subways...lol...Ryan is know getting a job there. **lol.** But anyways...I made lots of pizza and **it was fun**...later!

vader: {'neg': 0.046, 'neu': 0.556, 'pos': 0.398, 'compound': **0.9968**}

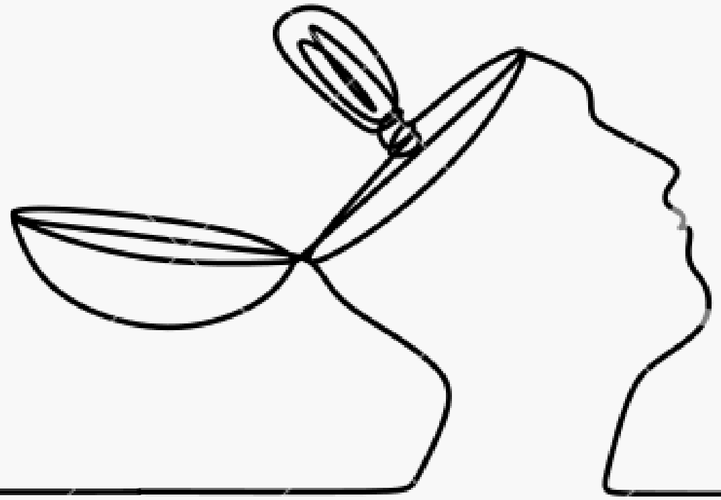


SENTIMENT ANALYSIS

blog 1:

..... **top five bad things** that have happened to me in the last week..... (not in order of importance) 1. i **sliced open my thumb** on the severed edge of the top of a cream of celery soup can. lots of blood. 2. i haven't had time to schedule a haircut. 3. i **didn't finish reading two papers** by my friday class last week. 4. i forgot to bring some stuff into school for my wife earlier today. 5. **something's wrong with my telephone** at home. What!? Is it my fault that I've got a good life? Did you *read* this blog last semester? Last semester **it sucked** to be me. I refuse to feel guilty for feeling ok about the world. (i'm sure i just **jinxed** everything though.) peace~

{'roberta_neg': **0.4675868**, 'roberta_neu': 0.3724617, 'reberta_pos': 0.15995146}

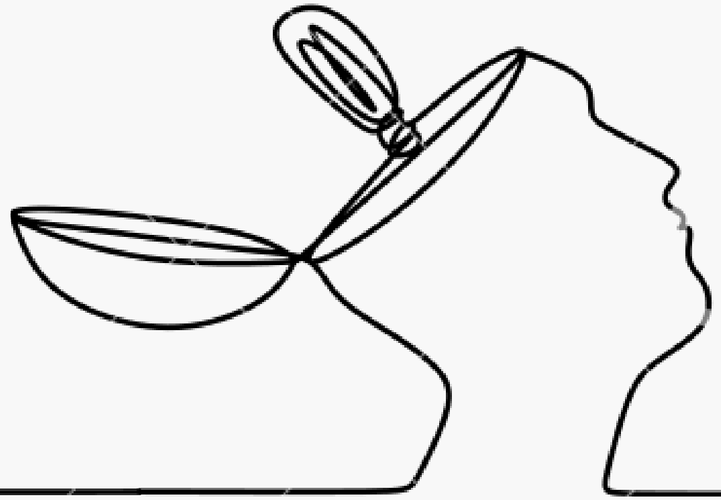


SENTIMENT ANALYSIS

blog 2:

Damn it's hump day. I was listening to the raadio today and they were talking about this event they're having comming up called Kegs and Eggs. It's a party from 8am-noon. It sounds like a lot of fun, but it's on a friday. It seems kinda **messed up** to take a whole day off just for a four hour party. And it seems like the think that you may stumble home from not walk. But Carbon Leaf is playing. I'll think about it, it's a month from friday (2 days prior to Jender's Birthday). On another note **I just ran out of money** (I mean until friday). **I kinda flipped out yesterday** about it. Then I realized I have another check before next rent. It's just with these alimony payments now, I have to rebudget. Yeah I know that's a lie, but I thought it was better than I spent $300 on a blanket and a pillow. Sometimes I just have to give myself a knowing look and say: 'Where's your head at?'

{'roberta_neg': **0.88296956**, 'roberta_neu': 0.09899507, 'reberta_pos': 0.018035335}

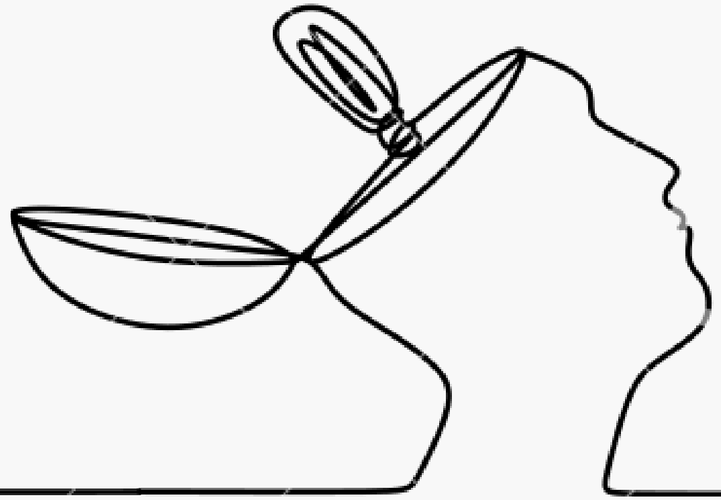


SENTIMENT ANALYSIS

blog 3:

Today was Ryan's sister's birthday. I was there at 9:30am and the party didnt start til 11:30am. But anyways I was there with Ryan. We hung out until the party and Ryan ate all the cheesecake and checked out the freshmen girls. Yeah **it was fun. Lol. I love Ryan he makes me laugh**! We had balloon fights...**I got into a fight** with Kyle and **that wasnt fun**. Well yeah **it was fun and I LOVE RYAN!** Today I went to work at Cazones.... **It was pretty fun** I met a new friend named Brandon. He thinks Iam beautiful. Lol. He told me when we were making subways...lol...Ryan is know getting a job there. **lol.** But anyways...I made lots of pizza and **it was fun**...later!

{'roberta_neg': 0.002418322, 'roberta_neu': 0.009020165, 'reberta_pos': **0.9885616**}



SENTIMENT ANALYSIS

apply vader and roberta to the whole dataframe

```
res = {}
for i, row in data.iterrows():
    try:
        text = row['text']
        id = row['blog_id']
        vader_score =sia.polarity_scores(text)
        vader_score_rename = {}
        for key, value in vader_score.items():
            vader_score_rename[f'vader_{key}'] = value
        roberta_score = roberta_polarity(text)
        both = {**vader_score_rename, **roberta_score}
        res[id] = both
    except Exception as e:
        print(f'broke for {id} {e}')
```

```
result_both = pd.DataFrame(res).T
result_both = result_both.reset_index().rename(columns = {'index':'blog_id'})
result_both = result_both.merge(data, how='left')
result_both.head()
```

assign labels according to score

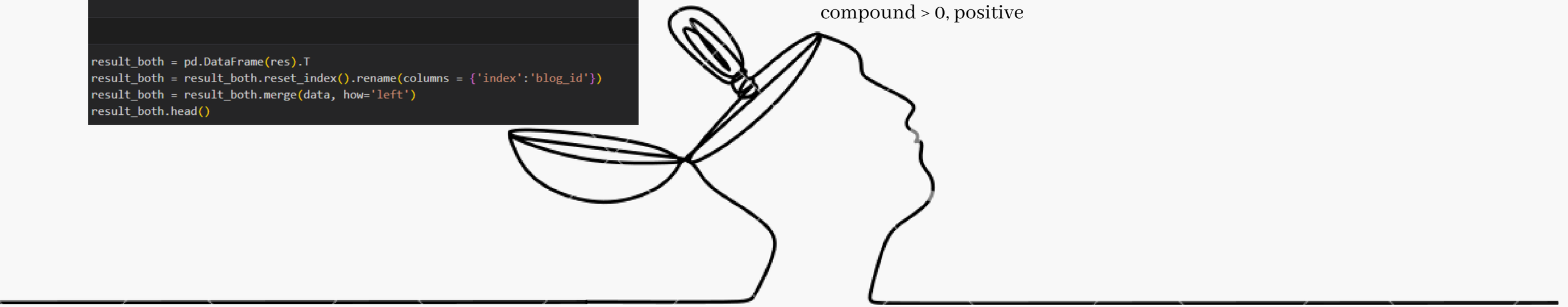
```
result_both.loc[:,'vader_cat'] = result_both['vader_compound'].\
    apply(lambda x : 'positive' if x > 0 else ('neutral' if x == 0 else 'negative'))
result_both.loc[:,'roberta_cat'] = result_both[['roberta_neg','roberta_neu','roberta_pos']].\
    idxmax(axis=1).apply(lambda x: 'positive' if x =='roberta_pos' \
                                    else ('neutral' if x == 'roberta_neu' else 'negative'))
```

Vader:

- compound < 0, negative
- compound = 0, neutral
- compound > 0, positive

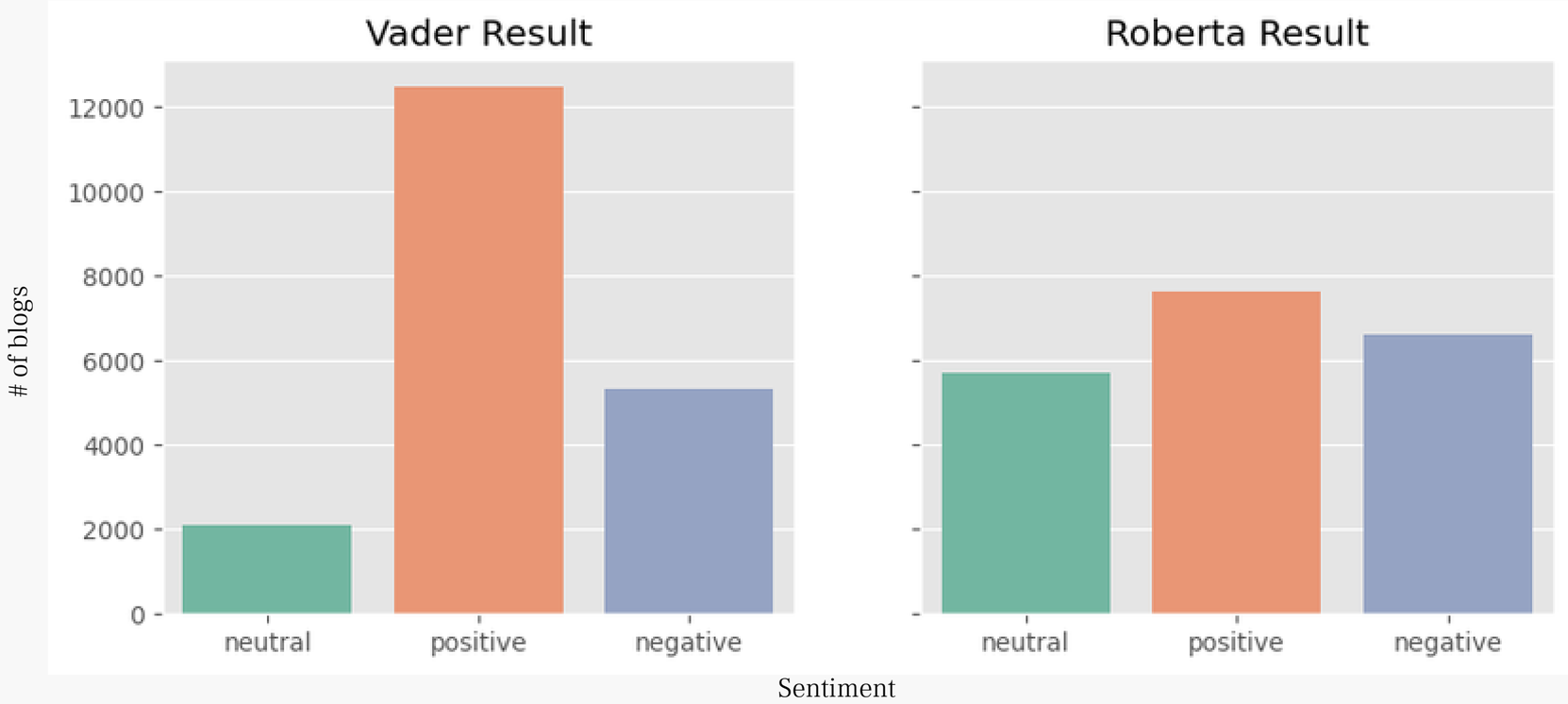
Roberta:

- each score is a probability
- tag being the one with max probability



SENTIMENT ANALYSIS

Model Comparison



Vader tends to label sentences as positive than neutral compared to Roberta

SENTIMENT ANALYSIS

Model Comparison - blog review

You know the reason why ... Oh fuck it. I have to fix all the other posts because once again I discovered something about the Internet that previously eluded me. Now that I get it I have to go through each one and fix it. Don't go snooping in the archives okay? Totally under construction. By the way, watched Quills tonight, had a chat with Been and felt totally depraved. I think the human spirit is awesome. Hug a friend today, Har, Har
[Vader: positive, Roberta: negative]

You know my chocolate sabbatical didn't go well.Will try again on Monday.More later!I have to clean my room(chantin')
[Vader : positive, Roberta: negative]

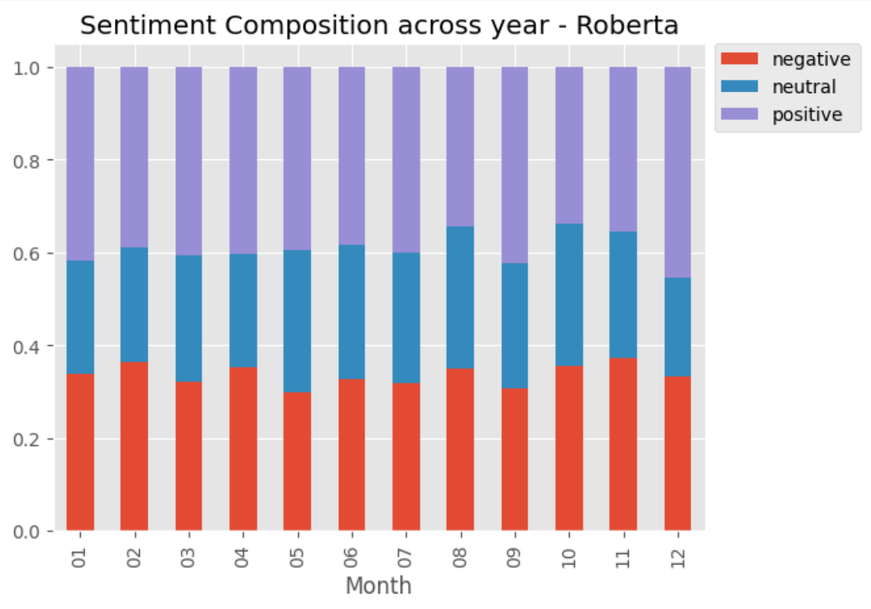
NOoooooo I can't get over it!
[Vader : neutral, Roberta: negative]

Such a boring day. Woke up late then didn't go for piano lesson, didn't go for service. Just stayed at home to rot. Practise piano, eat, sleep, study..so boring rite? ...
[Vader: positive, Roberta: negative]

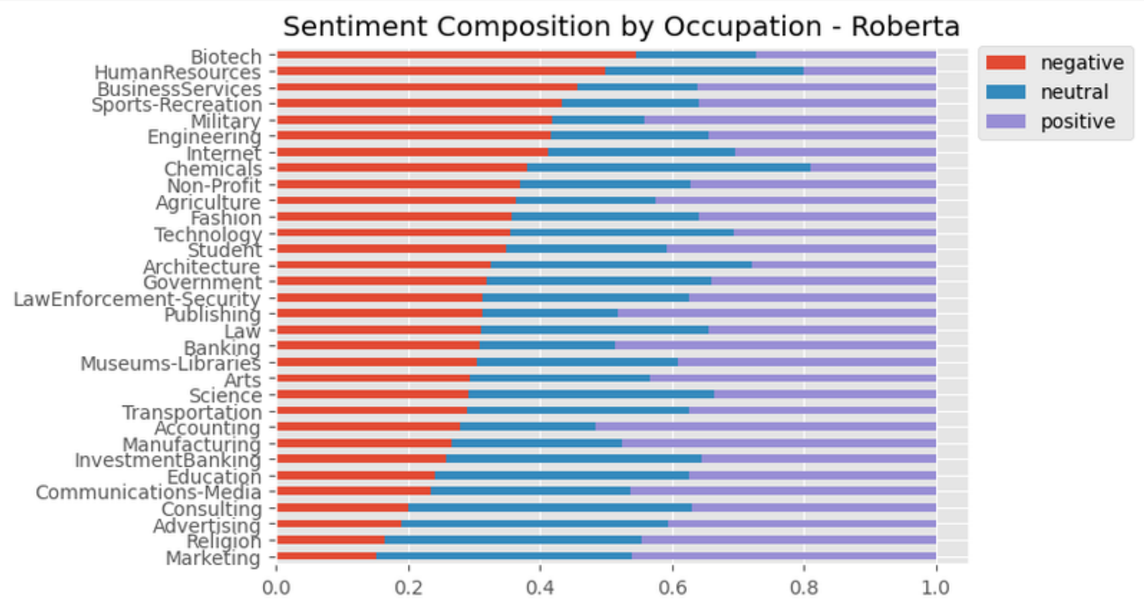
I'm going away now because I got bored of it. Check out the site for yourself, have fun.
[Vader : positive, Roberta: neutral]

SENTIMENT ANALYSIS

Findings



No significant relation between mood and month



Top 3 positive jobs: marketing, religion, advertising

Top 3 negative jobs: biotech, HR, business services

AUTHOR PROFILING

AGE & GENDER PREDICTION

- data preparation
- model comparison
- what can we do with the result ?

AUTHOR PROFILING

AGE & GENDER PREDICTION

DATA CLEANING

20000 blogs used due to memory constraint for Naive Bayes

blogs with 5-200 words

encoded gender and age

* cleaned text used

```
data_raw = pd.read_csv('cleaned_text_dump.csv')
data_raw = data_raw[(data_raw['word_count']>5) & (data_raw['word_count']<200)]
data_raw = data_raw.dropna()
data_raw.loc[:,'gender'] = [1 if x =='female' else 0 for x in data_raw['gender']]
data_raw.loc[:,'age_coded'] = np.where(data_raw['age']<20,0,
                                       np.where(data_raw['age']<30,1,
                                       np.where(data_raw['age']<40,2,3)))
data_raw.reset_index(drop = True, inplace=True)
data_ml_sm = data_raw[:20000]
```

AUTHOR PROFILING

GENDER PREDICTION

Naive Bayes

```
X = data['clean_text']
vectorizer = CountVectorizer(max_features = 50000,stop_words='english', ngram_range=(1,3))
sparce_matrix = vectorizer.fit_transform(X)
X = sparce_matrix.astype(np.uint8).toarray()
y = data['gender'].astype(np.uint8)
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2, random_state=42)
nb = GaussianNB()
nb.fit(X_train,y_train)
y_pred = nb.predict(X_test)
print(classification_report(y_test, y_pred))
```

NB:

n-gram = (1,3), test size = 0.2

acc: 0.664

LR:

n-gram = (1,2), test size = 0.2

acc: 0.755

Logistic Regression

```
X = data['clean_text']
vectorizer = CountVectorizer(stop_words='english', ngram_range=(1,2))
y = data['gender'].astype(np.uint8)
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
x_train = vectorizer.fit_transform(x_train)
x_test = vectorizer.transform(x_test)
clf = LogisticRegression()
clf.fit(x_train, y_train)
y_pred = clf.predict(x_test)
print('Logistic Regression accuracy for gender prediction: ',metrics.accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

AUTHOR PROFILING

AGE PREDICTION

Naive Bayes

```
X = data['clean_text']
vectorizer = CountVectorizer(max_features = 50000,stop_words='english', ngram_range=(1,3))
sparce_matrix = vectorizer.fit_transform(X)
X = sparce_matrix.astype(np.uint8).toarray()
y = data['age_coded'].astype(np.uint8)
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2, random_state=42)
nb = GaussianNB()
y_pred = nb.fit(X_train,y_train).predict(X_test)
print(classification_report(y_test, y_pred))
```

NB:

n-gram = (1,3), test size = 0.2

acc: 0.594

LR:

n-gram = (1,3), test size = 0.2

acc: 0.654

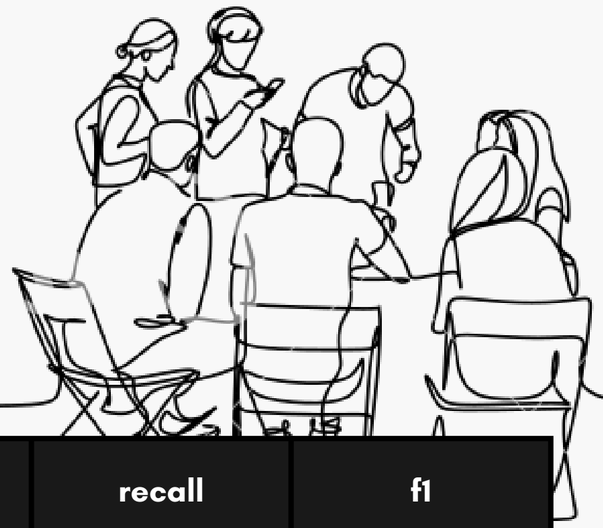
Logistics Regression

```
X = data.clean_text
y = data.age_coded
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state=42)
vectorizer = CountVectorizer(max_features = 50000,stop_words='english', ngram_range=(1,3))
x_train = vectorizer.fit_transform(x_train)
x_test = vectorizer.transform(x_test)

clf = LogisticRegression(multi_class='multinomial' , solver = 'lbfgs')
y_pred = clf.fit(x_train, y_train).predict(x_test)
print('Logistics Regression accuracy for age prediction: ', metrics.accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

AUTHOR PROFILING

AGE & GENDER PREDICTION



	precision		recall		f1	
	NB	LR	NB	LR	NB	LR
male	0.77	0.76	0.63	0.86	0.69	0.81
female	0.56	0.74	0.71	0.59	0.63	0.66
acc					0.66	0.76

	precision		recall		f1	
	NB	LR	NB	LR	NB	LR
10s	0.62	0.72	0.71	0.74	0.66	0.73
20s	0.60	0.63	0.56	0.65	0.58	0.64
30s	0.58	0.61	0.49	0.58	0.53	0.60
40s	0.18	0.45	0.22	0.16	0.20	0.23
acc					0.59	0.66

AUTHOR PROFILING

WHAT'S THE APPLICATION ?

AUTHOR PROFILE:

marketing

forensic linguistics

TEXT CLASSIFICATION:

cyber bullying detection

harassment detection (pedophile threads etc.)

TOPIC ANALYSIS

WORD FREQUENCY

import libraries

```
from nltk.probability import FreqDist
from nltk.tokenize import word_tokenize
```

divide age groups

```
blog_10s = data[data['age_coded']==0]
blog_20s = data[data['age_coded']==1]
blog_30s = data[data['age_coded']==2]
blog_40s = data[data['age_coded']==3]
```

build word frequency dictionary

```
sentence_10s = blog_10s['clean_text'].tolist()
fdist_10s = FreqDist()
for sentence in sentence_10s:
    for word in word_tokenize(sentence):
        if word not in to_remove and len(word )>1:
            if word in fdist_10s.keys():
                fdist_10s[word] += 1
            else:
                fdist_10s[word] = 1

fdist_10s_raw = fdist_10s.most_common(100)
print(fdist_10s_raw)

vague_word_10s = ['like', 'know', 'think', 'day', 'today', 'good', 'time', 'thing', 'come', 'want',
                  'need', 've','feel', 'look','find','word','tell','new','way','start','try', 'leave',
                   'mean', 'long', 'year', 'let','year','little','use','happen','end','hour', 'ask',
                   'com', 'na','wan', 'yesderday', 'today', 'sit', 'hear', 'bit','post', 'oh' ,'love',
                   'work','life','people', 'blog','th']
for i in vague_word_10s:
    try:
        fdist_10s.pop(i)
    except KeyError:
        continue
```

TOPIC ANALYSIS

TOPIC & VOCABULARY

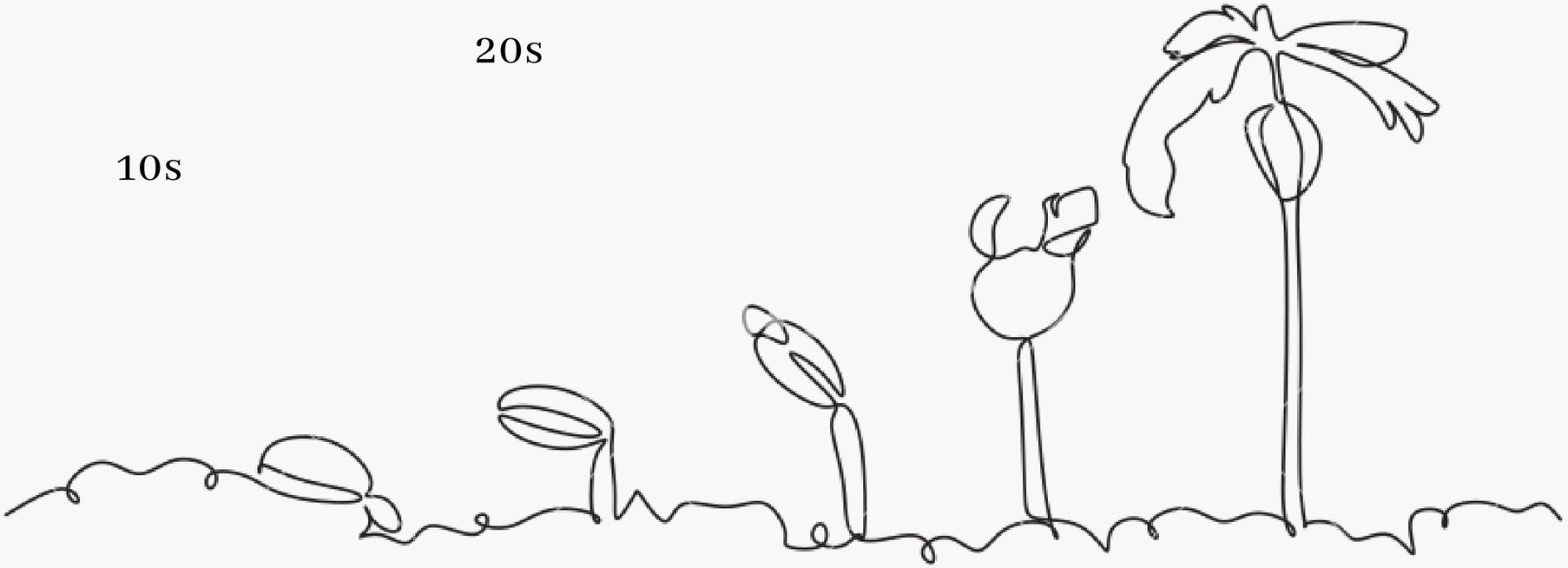
Do we talk about different things as we grow old ?

40s

30s

20s

10s

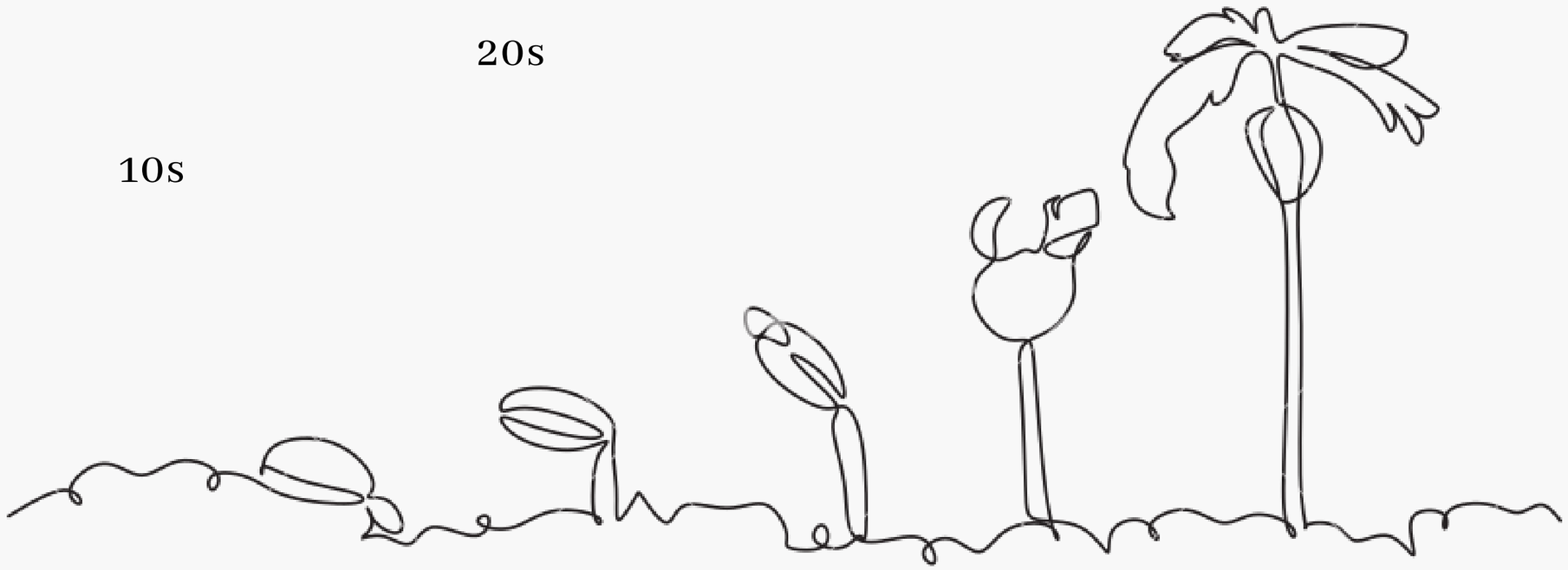


TOPIC ANALYSIS

TOPIC & VOCABULARY

Do we talk about different things as we grow old ?

YES !



10s

Topic

School life related:
school, test, class,
book, study ...

mom & dad

play, girl, bank, game

Vocacbulary

Young people
slang:lol, yay, ya, cuz
kinda, hehe, haha,

and ?

a lot of swear words !
fuck, damn, shit, hell,
suck ...

Mood

happy, fun, great

but also

boring & bored, sad,
suck, stupid, hate,
tired

20s

Topic

still school but less:
school, class, course

work related:
job, email
also:
home, house, couple, party, sleep, money..

Vocacbulary

Yound people slang:
much less

swear words:
yes but less

Mood

happy, fun,

but also
boring & bored, sad, suck

30s

Topic

School, job, church, house, kid

and ?

politics shows up ! iraq, john, bush, war, president, news ..

Vocacbulary

more formal and almost no swear words

Mood

less words with emotions

happy, hope, fun

also hate (small one!)

40s

Topic

A lot more politics!
Iraq war, news, president, bush, news, american, china, energy, power, church

home, friend, job, family

Vocacbulary

not very different from the 30s
formal
project, article, report

Mood

not obvious, positive in general

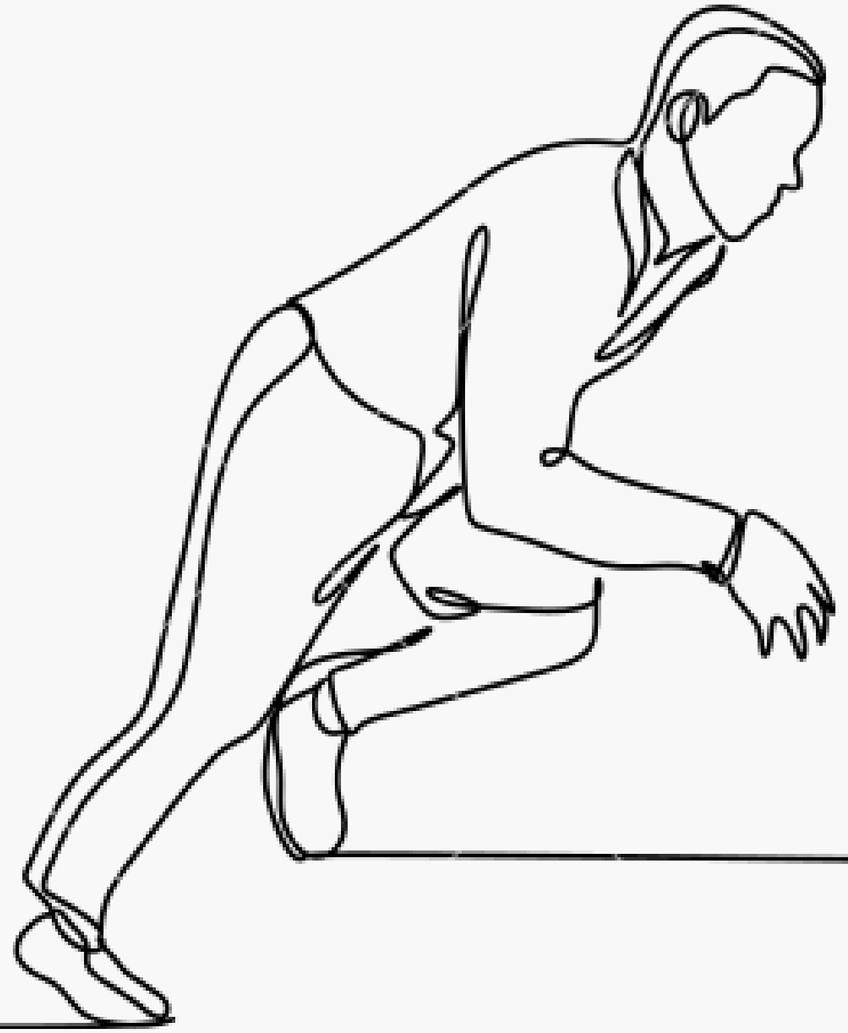
LIMITATIONS & IMPROVEMENTS

- bigger sample size for machine learning , upsample the older age group

- text classification: tf-idf in addition to Countvectorizer

  RNN (Lexicon-Based Supervised Attention Model)

- topic analysis : topic modeling & topic classification



Q & A