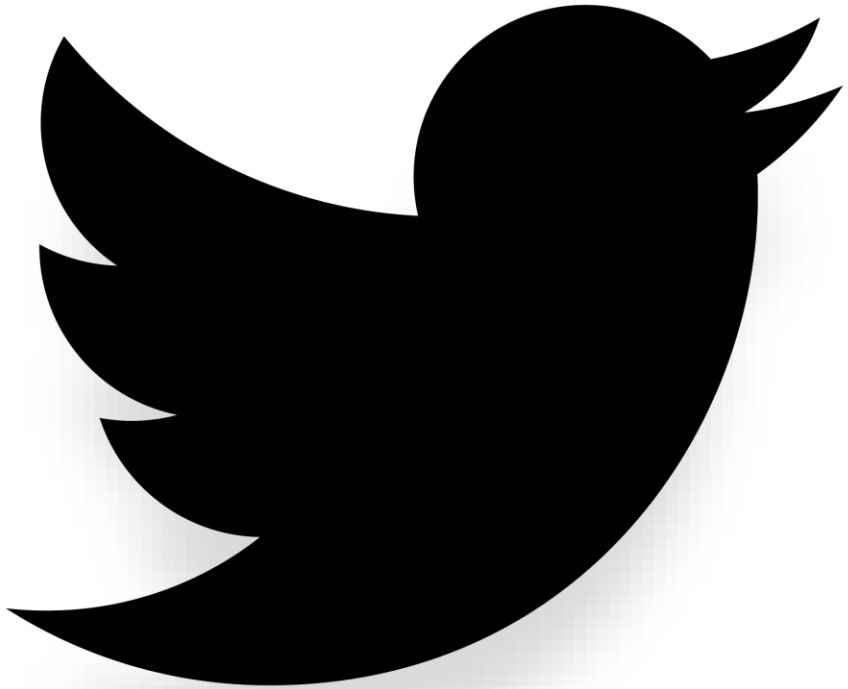
AI-Assisted Tool for Marketing Exploration

Group 3

- Ashna Prasannan
- Jiya Peter
- Najna Nazeer
- Rincy Jose
- Duc Anh Trinh



Motivation

- There are over one in five adults in US are active on Twitter. Twitter is the most popular platform among social media networks that people used for exchange of daily information.

- Twitter is a valued channel for marketing analysis

Ideas

Aim:

Build an AI-based platform for **real-time and comprehensive twitter analysis** that provides a useful information about a brand, for example, customer attitude trends over specific time or in a certain nation.

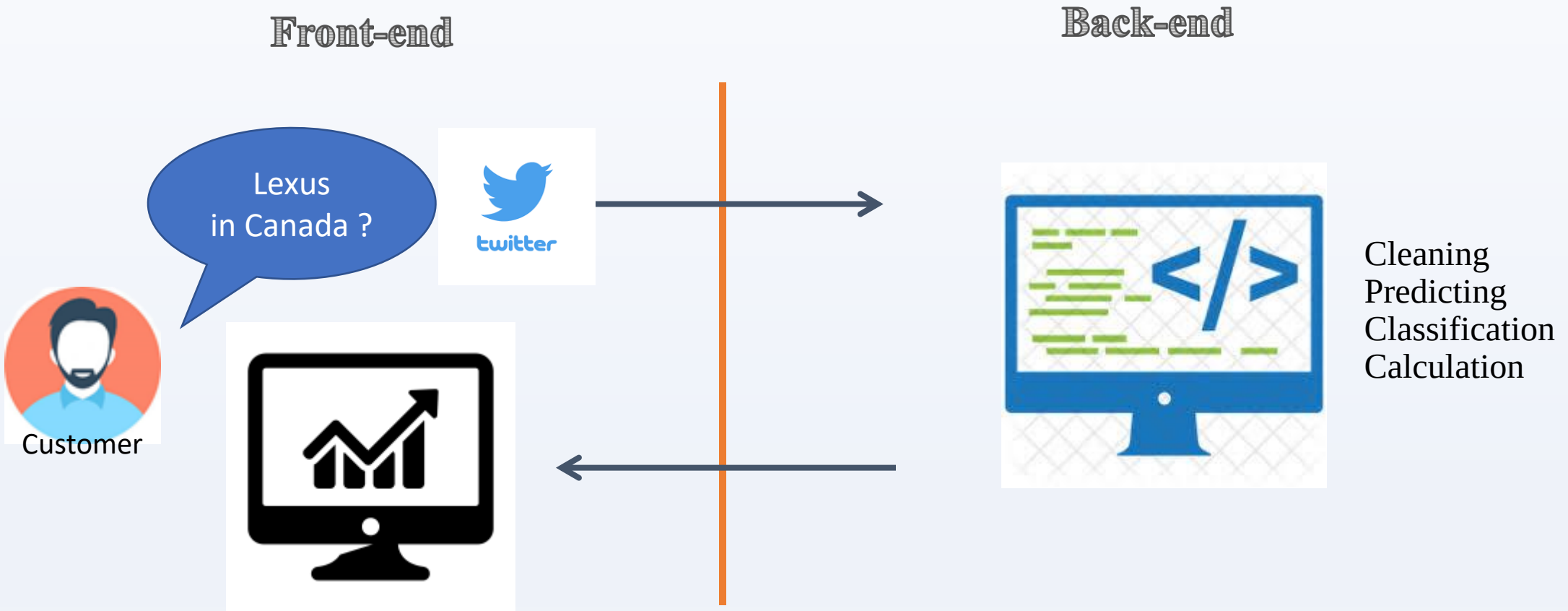
Main Tasks:

Task 1: **Training** an “emotional” **classifier** based on a human-based prelabeled data set, which can distinguish a positive, neutral or a negative tweet.

Task 2: Using a Twitter API to **collect tweets** about an interested product. Evaluate sentiment with classifier. Analyze classified tweets to get insights

Task 3: **Creating Web UI** using Django web framework and embedding the model.

Systemic view

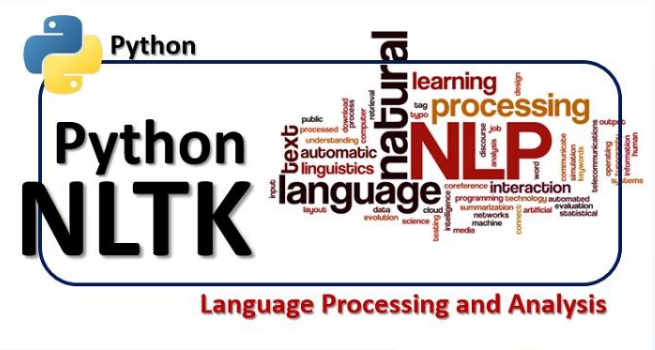


Tools

- Twitter Collection
  - Tweepy
- Data Pre-processing
  - NLTK
  - NumPy
  - Pandas
- Data Visualization
  - Matplotlib
  - Pandas plotting

- Modeling:
  - Sci-kit learn
  - Tensor flow
  - Keras
  - Jolib

- Clouding deployment:
  - Django
  - Django REST Framework
  - Html, bootstrap
  - Pythonanywhere



Part 1:
Dataset and Preprocessing

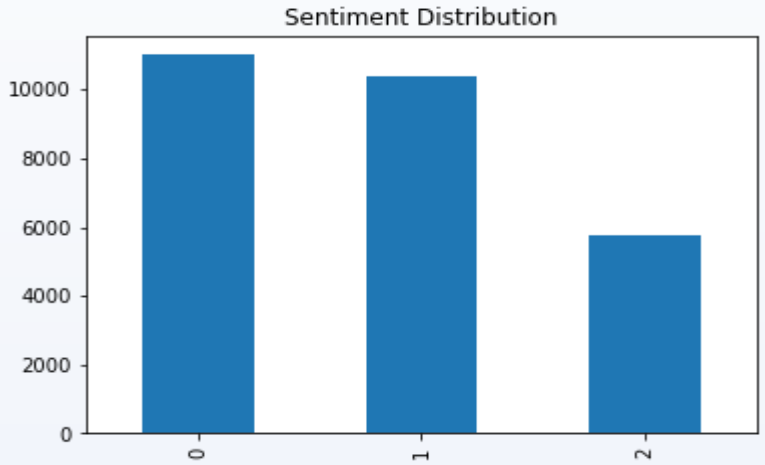
Dataset

- ***Semeval:*** The SemEval [27] corpus is formed by 5232 positive tweets and 2067 negative tweets annotated by human evaluators using the crowdsourcing platform Amazon Mechanical Turk.
- ***6HumanCoded***. The 6HumanCoded dataset is a collection of 1340 positive and 949 negative tweets scored according to positive and negative numeric scores by six human evaluators.
- **Sanders**. The Sanders dataset consists of 570 positive and 654 negative tweets evaluated by a single human annotator.
- **Twitter US Airline Sent**: Twitter data was scraped from February of 2015 and contributors were asked to classify positive, negative, and neutral tweets.

Dataset summary

```
tweet_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27152 entries, 0 to 27151
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Tweet   27152 non-null  object
 1   Sent    27152 non-null  int64 
dtypes: int64(1), object(1)
memory usage: 424.4+ KB
```




```
tweet_df.iloc[::500, :]
```

	Tweet	Sent
0	@VirginAmerica What @dhepburn said.	1
500	@VirginAmerica still waiting to see @Starryeyes_Dev_ 😔	1
1000	@united how can you not put my bag on plane to Seattle. Flight 1212. Waiting in line to talk to someone about my bag. Status ...	0
1500	@united I sent a message. It's very irresponsible that my suitcase cannot be found. This is truly worst service ever... #fedup...	0

Clean dataset

```
def remove_stopwords(sentence):
    '''
    function to remove stopwords
```

```
def lemmatize(sentence):
    '''
    function to do lemmatization
```

Embedding

```
def load_glove_model(glove_file):
    print("[INFO]Loading GloVe Model...")
    model = {}
```

```
def sent_vectorizer(sent, model):
    '''
    sentence vectorizer using the pretrained glove model
    '''
```

```
def build_model(nb_words, rnn_model="SimpleRNN", embedding_matrix=None):
    '''
    build_model function:
```

Modeling

- SimpleRNN
- LSTM
- GRU

Tweet Analysis

	Text	Tweet_punct	clean_tweets	Tweet_tokenized
0	b"RT @rutoogenre: He's still smart now, during tmap golden bell challenge he's the only who got all the questions correct till the end / he g\xe2\...	bRT rutoogenre Hes still smart now during tmap golden bell challenge hes the only who got all the questions correct till the end he gxexxa	still smart now during golden bell challenge the only who got all the correct till the end he	still smart now during golden bell challenge the only who got all the correct till the end
1	b"@MrGee54 She's looking for Deion's son since she has history with JSU QBs..."	bMrGee Shes looking for Deions son since she has history with JSU QBs	looking for son since she history with	looking for son since she history with
2	b'@Inkslasher I have theory that bell is still alive, it\xe2\x80\x99s weird but hear me out, in the cod mobile 2nd anniversary\xe2\x80\xa6 https:...	bInkslasher I have theory that bell is still alive itxexxs weird but hear me out in the cod mobile nd anniversaryxexxa	I have theory that bell is still alive weird but hear me out in the cod mobile	have theory that bell still alive weird but hear out the cod mobile

Embedding with GloVe

GloVe: pre-trained word vectors that is trained on 2 billion tweets, which contains 27 billion tokens, 1.2 million vocabs.

```
def load_glove_model(glove_file):
    print("[INFO]Loading GloVe Model...")
    model = {}
    with open(glove_file, 'r', encoding="utf8") as f:
        for line in f:
            split_line = line.split()
            word = split_line[0]
            embeddings = [float(val) for val in split_line[1:]]
            model[word] = embeddings
    print("[INFO] Done...{} words loaded!".format(len(model)))
    return model
nlp = spacy.load("en_core_web_sm")
```

```
[INFO]Loading GloVe Model...
[INFO] Done...1193514 words loaded!
Found 39676 unique tokens.
```

Modeling with RNN

Simple RNN	LSTM	GRU
SimpleRNNs are good for processing sequence data for predictions but suffers from short-term memory	• mitigate short-term memory using mechanisms called gates. • Slow	mitigate short-term memory using mechanisms called gates. Faster than LSTM (as a shorthand version of LSTM)

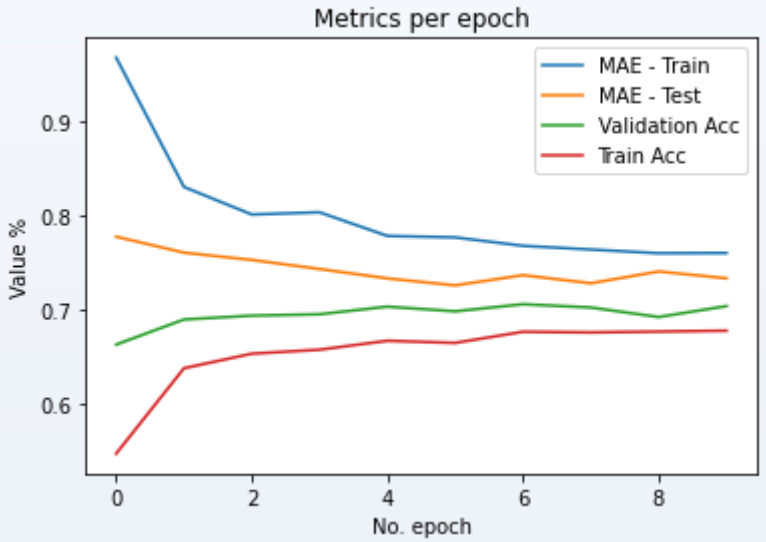
```
if rnn_model == "SimpleRNN":
    model.add(SimpleRNN(200, dropout=0.2, recurrent_dropout=0.2))
elif rnn_model == "LSTM":
    model.add(LSTM(200, dropout=0.2, recurrent_dropout=0.2))
else:
    model.add(GRU(200, dropout=0.2, recurrent_dropout=0.2))
  model.add(Dense(500,activation='relu'))
# model.add(Dense(500, activation='relu'))
model.add(Dense(3, activation='softmax'))

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

Modeling

```
model_rnn = build_model(nb_words, "SimpleRNN", embedding_matrix)
history = model_rnn.fit(train_X, train_y, epochs=20, batch_size=300,
          validation_data=(valid_X, valid_y), callbacks=[EarlyStopping(monitor='val_accuracy'
predictions = model_rnn.predict(valid_X)
predictions = predictions.argmax(axis=1)
print(classification_report(valid_y, predictions))
```

	precision	recall	f1-score	support
0	0.71	0.85	0.78	2203
1	0.70	0.64	0.67	2057
2	0.68	0.55	0.60	1171
accuracy			0.70	5431
macro avg	0.70	0.68	0.68	5431
weighted avg	0.70	0.70	0.70	5431



Comparison

SimpleRNN

	precision	recall	f1-score	support
0	0.71	0.85	0.78	2203
1	0.70	0.64	0.67	2057
2	0.68	0.55	0.60	1171
accuracy			0.70	5431
macro avg	0.70	0.68	0.68	5431
weighted avg	0.70	0.70	0.70	5431

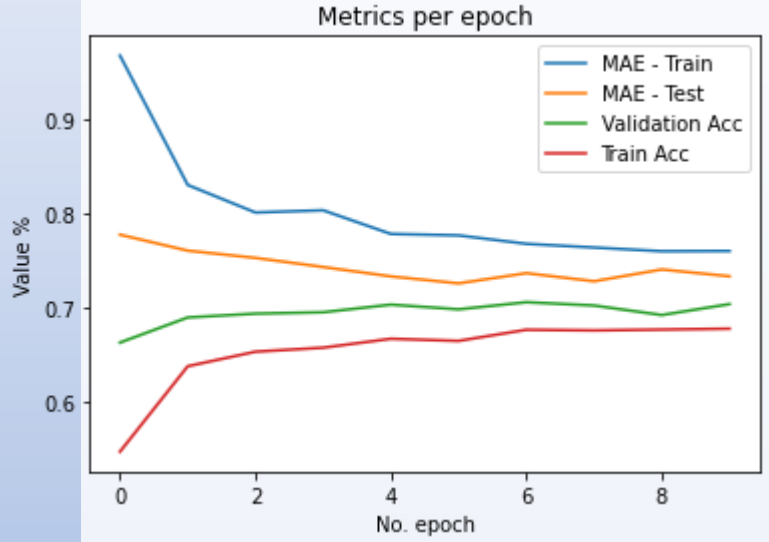
LSTM

	precision	recall	f1-score	support
0	0.79	0.80	0.80	2203
1	0.70	0.73	0.72	2057
2	0.68	0.62	0.65	1171
accuracy			0.74	5431
macro avg	0.73	0.72	0.72	5431
weighted avg	0.74	0.74	0.74	5431

GRU

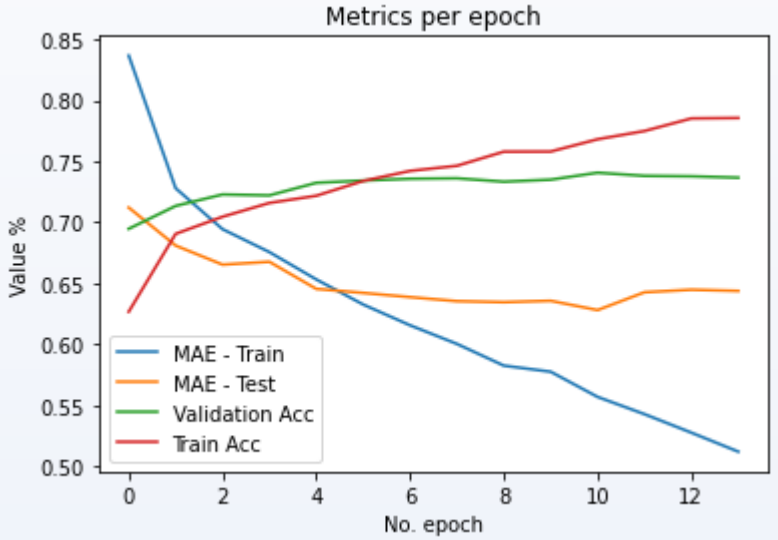
	precision	recall	f1-score	support
0	0.74	0.87	0.80	2203
1	0.75	0.65	0.69	2057
2	0.67	0.61	0.64	1171
accuracy			0.73	5431
macro avg	0.72	0.71	0.71	5431
weighted avg	0.73	0.73	0.72	5431

Comparison



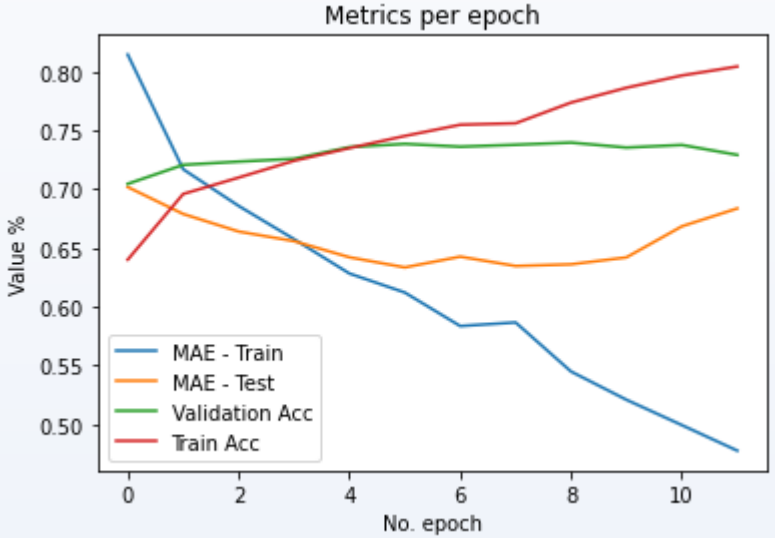
SimpleRNN

Accuracy: 0.70



LSTM

Accuracy: 0.74



GRU

Accuracy: 0.73

Overfitting Reduction

- Stop-words
- Lemmatization
- Remove mention
- Remove url, non-letter, digit

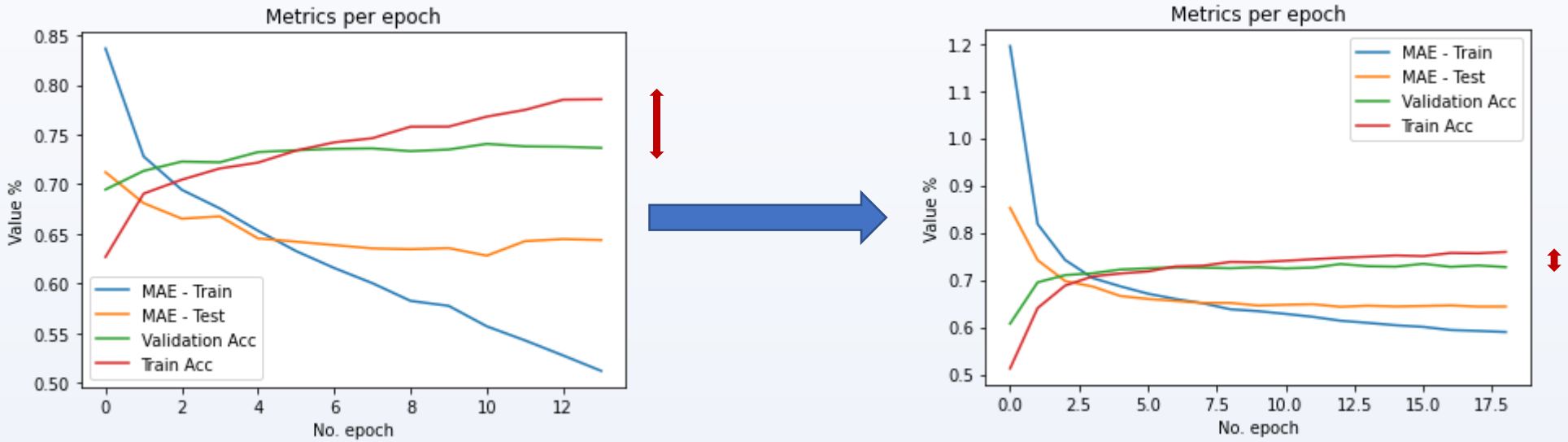
```
def clean_fnt(text):
    new = []
    sentence = nlp(text)
    for tk in sentence:
        if tk.is_stop == False:
            new.append(tk)

    c = " ".join(str(x) for x in new)
    doc = nlp(c)
    s = ''
    for token in doc:
        s +=" " + token.lemma_

    tweet = re.sub(r'@\w+',' ', s)
    tweet = re.sub(r'http\S+', ' ', tweet)
    tweet = re.sub(r'\W+', ' ', tweet)
    tweet = re.sub('\w*\d\w*', ' ', tweet)
    return tweet
```

Tweet	Sent	cleaned_Tweet
@VirginAmerica What @dhepburn said.	1	say
@VirginAmerica plus you've added commercials to the experience... tacky.	2	plus add commercial experience tacky

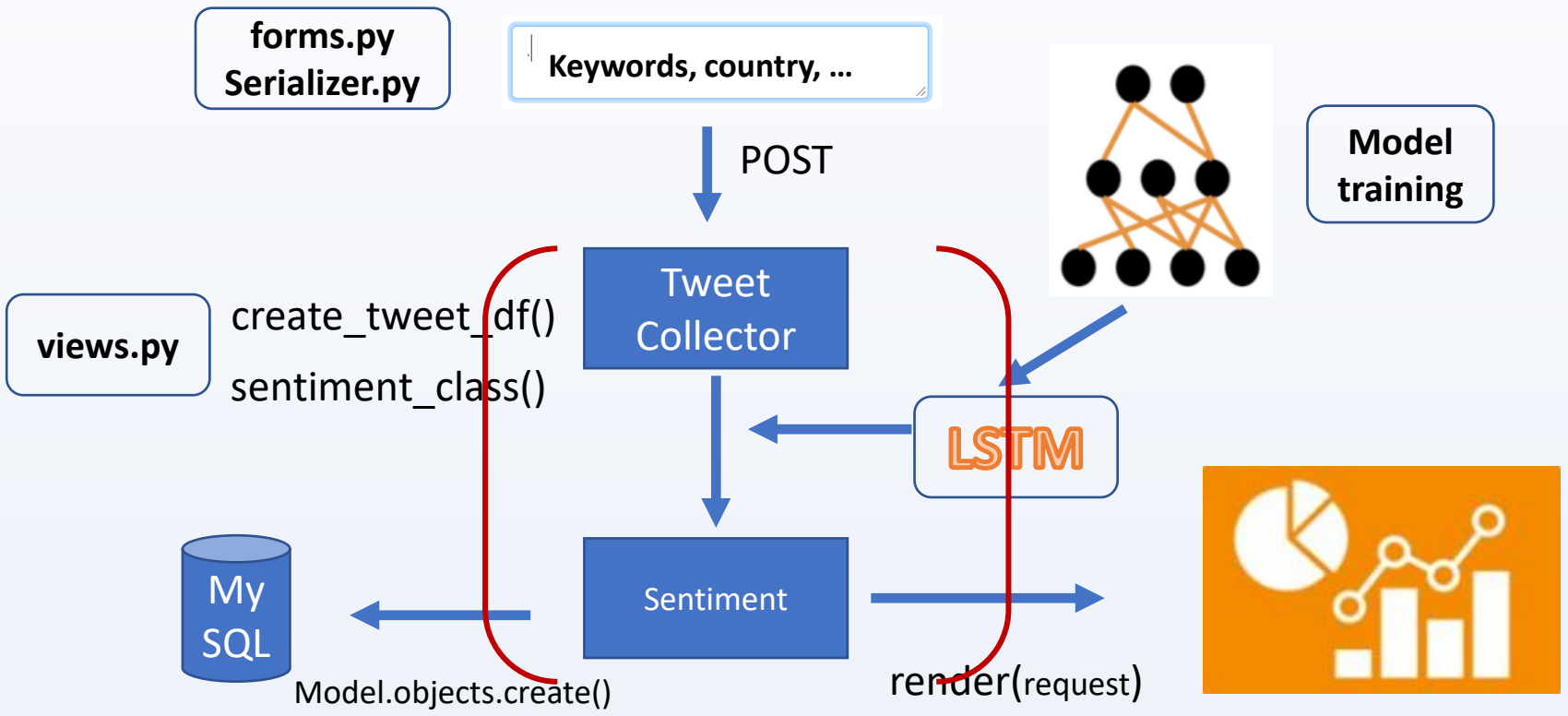
Overfitting Reduction



model_lstm.save('my_model_lstm2.h5')

Part 2:
Tweet Collector

Tweet Collector

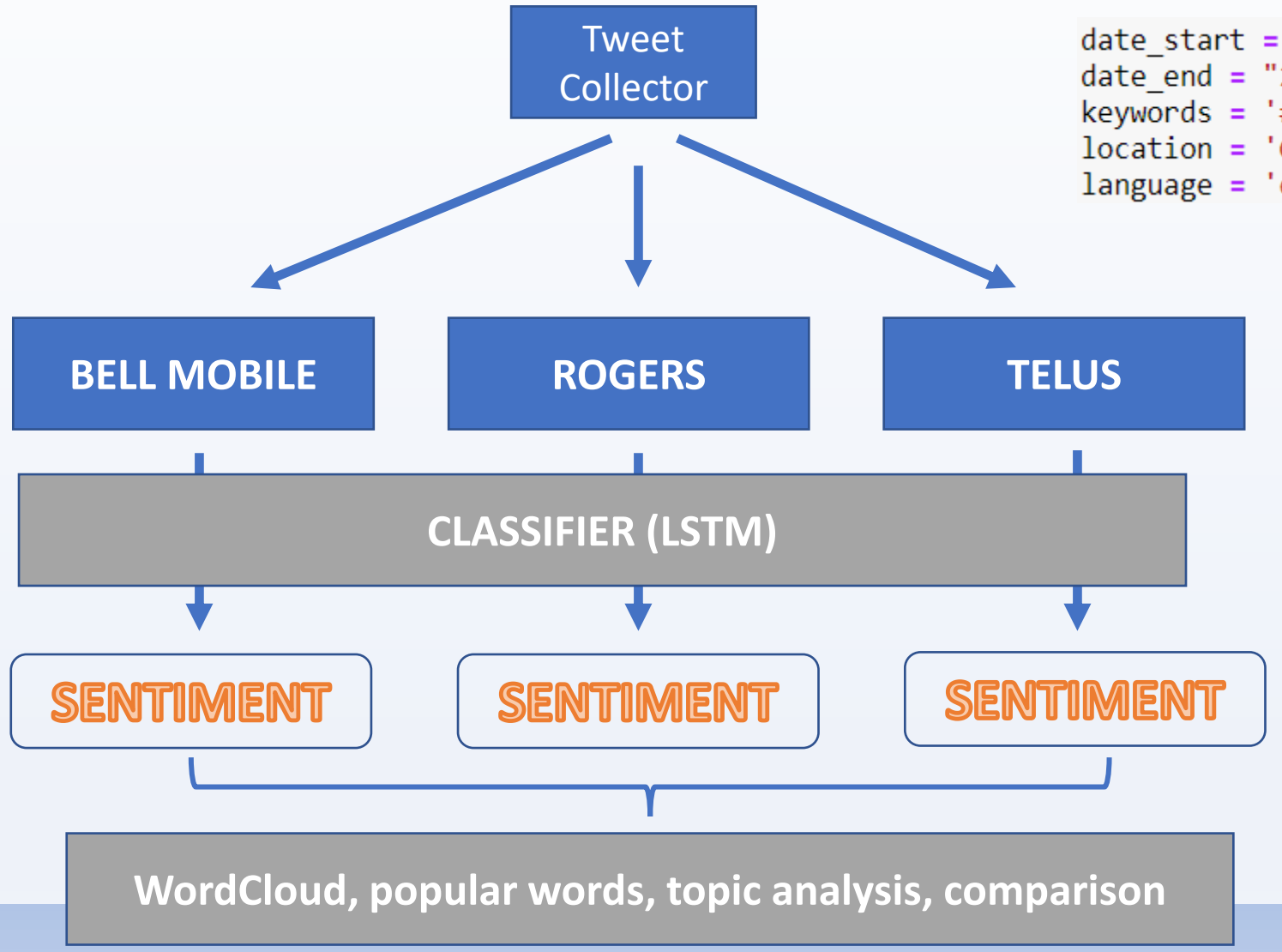


Tweet Sentiment Prediction


```
date_start = "2021-02-25"
date_end = "2021-03-04"
keywords = 'BELL AND mobile OR bell OR Bell'
location = 'Canada'
language = 'en'

# Get country id
places = api.search_geo(query=location,granularity="country")
place_id = places[0].id
```

Sentiment Prediction




```
date_start = "2021-02-25"
date_end = "2021-03-04"
keywords = '#BELL AND mobile OR bell OR Bell'
location = 'Canada'
language = 'en'
```

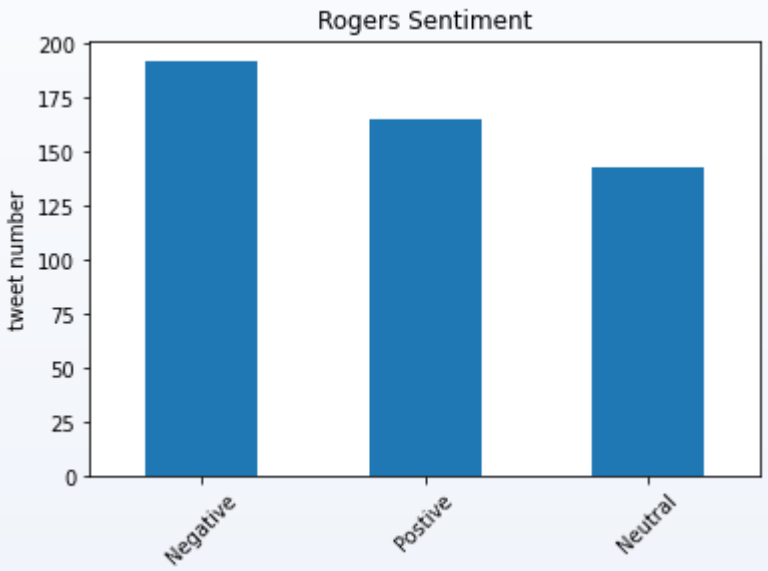
Sentiment Prediction

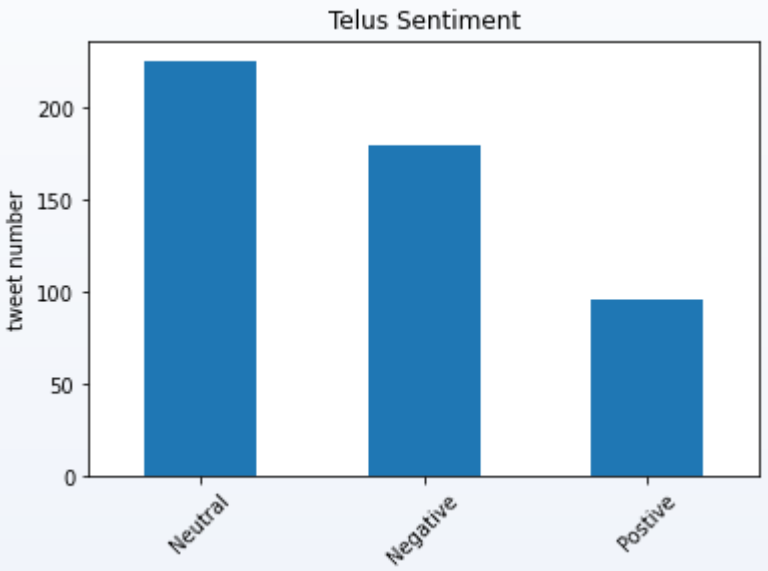
"b'RT @LeeDForster: 18 minutes into @SkySportsNews at 6pm and they\xe2\x80\x99re still talking about #NUFC\n\nKlopp, Gerrard, Rafa, Rogers\xe2\x80..."

```
array([[ 42, 65, 1385, 1186, 1168, 554, 556, 557, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]])
```

PREDICTED SENTIMENT: **negative**
Probability: **0.9997768998146057**

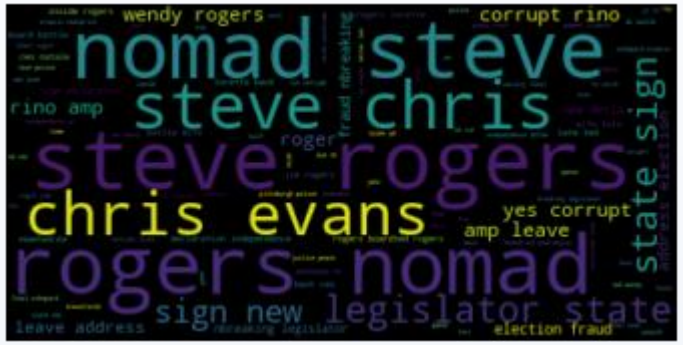
Sentiment Distribution

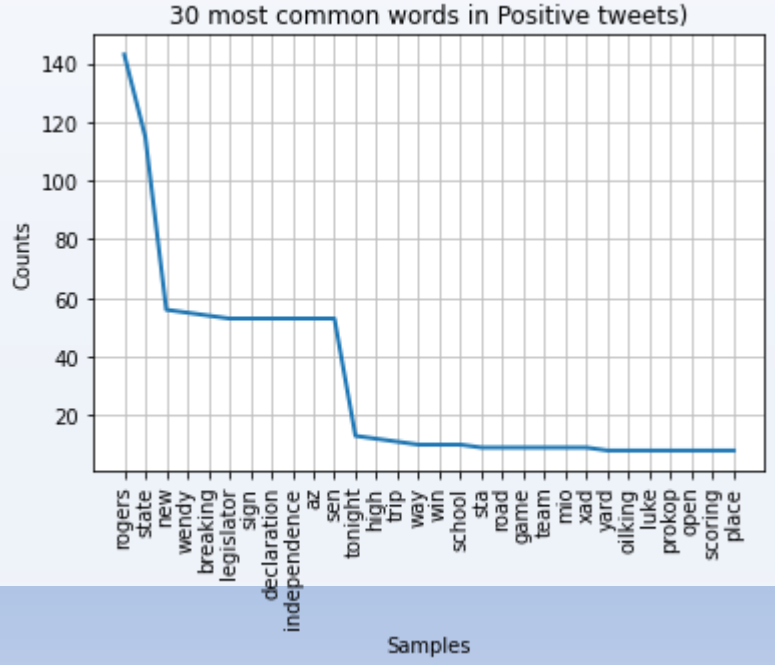




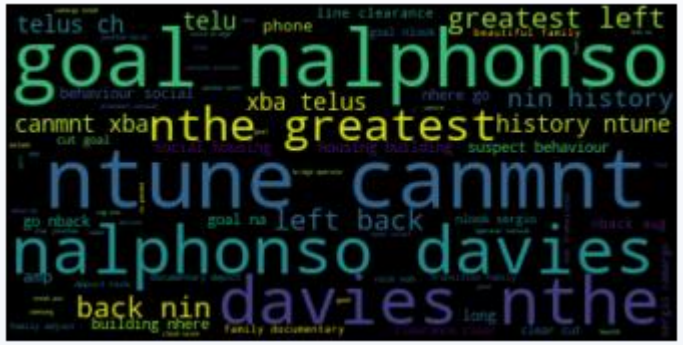
Visualization

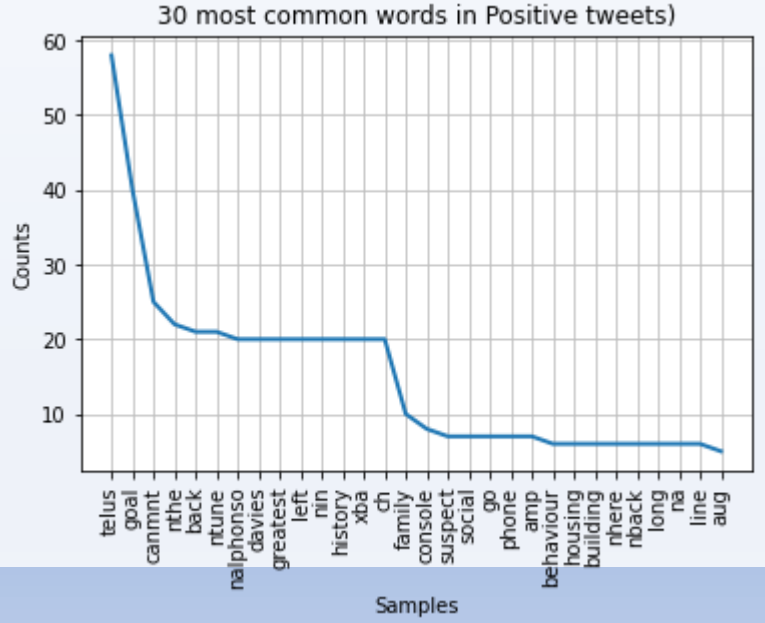
Rogers





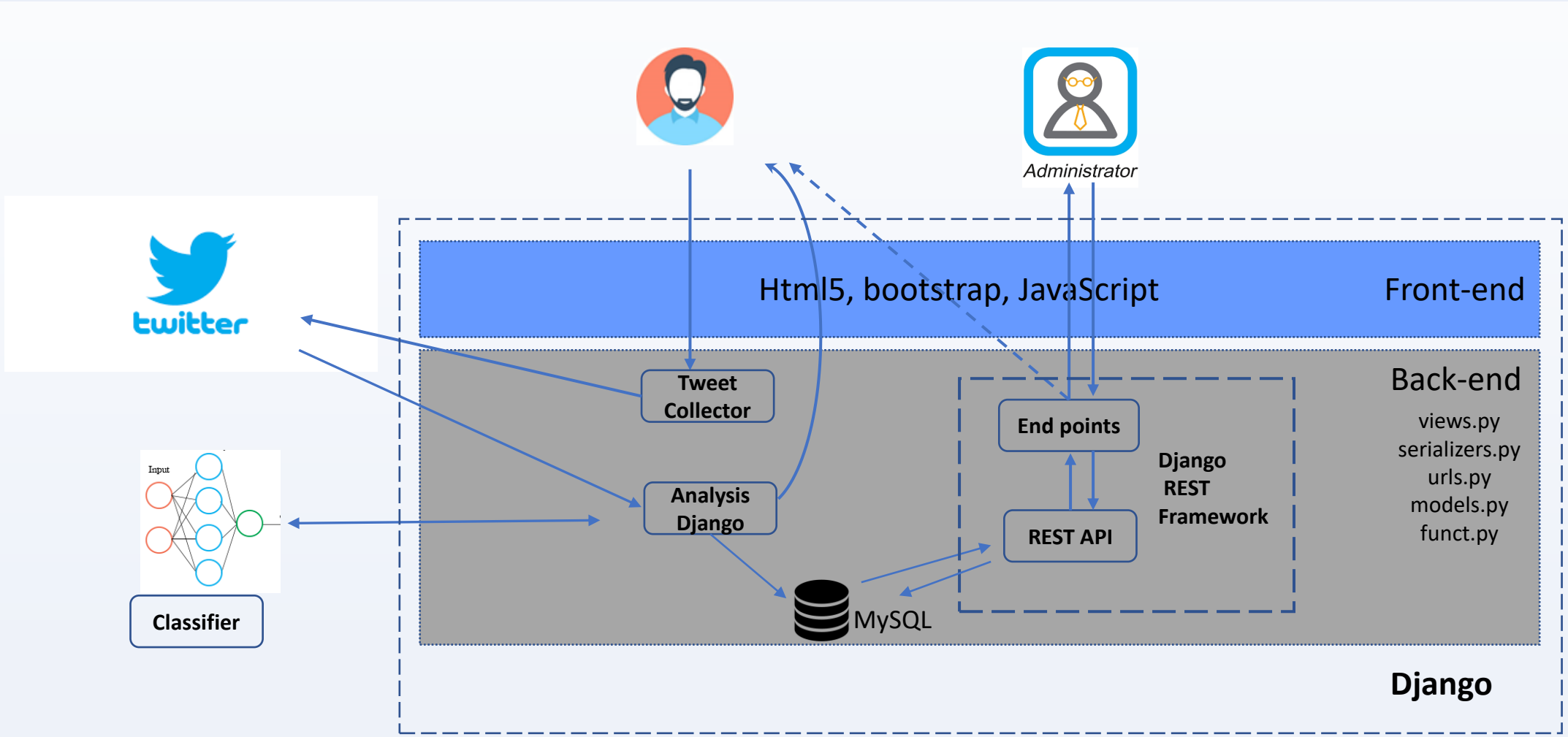
Telus





Part 3:
Django Web Framework
& Clouding deployment

TweetAnalyz

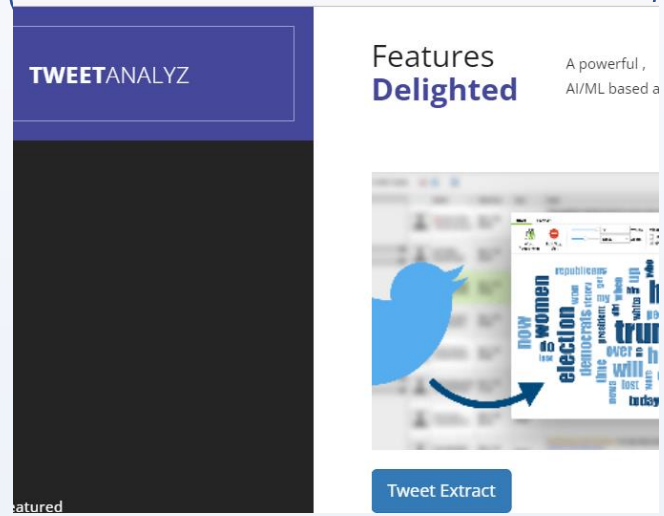


Deployment

GitHub repository

git clone …

http://aml3406.pythonanywhere.com/

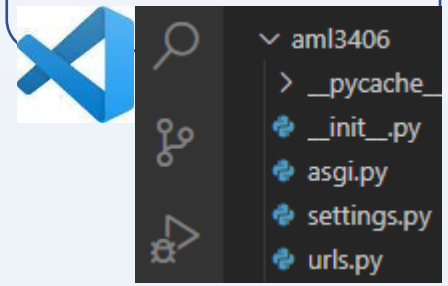


git pull …

git push …

Jupiter Notebooks

Local folders



Conclusion and Future Scope

- Implemented an AI-based real-time twitter sentiment analyzer for marketing exploration
- Deployed it on cloud-PythonAnywhere
- Analysis and visualization of negative tweets using this sentiment analyzer will give a hint for their brand that which aspect/feature they have to improve to enhance their profit.
- Beneficial for any firm to analyze and evaluate the public sentiment towards their product/brand.


THANK
YOU
GRACIAS
ARIGATO
SHUKURIA
BOLZÏN
MERCI
BÏYAN
SHUKRIA
DANKSCHEEN
TASHAKKUR ATU
SUKSAMA
MEHRBANI
GRAZIE
JUSPAXAR