

MULTILINGUAL EMOJI PREDICTION

PROBLEM DESCRIPTION

Since there is an importance of visual icons for providing an additional layer of meaning to social media messages and the indisputable role of Twitter as one of the most important social media platforms, so we have to design a system to predict whether given a tweet is in English or Spanish and its most likely associated emoji. For accomplishing this, we will remove the emoji from the tweet and ask users to predict it (ignoring tweets including more than one emoji).

Given the paramount importance of visual icons for providing an additional layer of meaning to social media messages, on one hand, and the indisputable role of Twitter as one of the most important social media platforms, on the other, we propose the Emoji Prediction task. We will challenge systems to predict emojis among a wide and heterogeneous emoji space. As for the experimental setting, we will remove the emoji from the tweet and ask users to predict it, ignoring tweets including more than one emoji, for simplicity purposes. We will provide data for the two tasks:

- Subtask 1: Emoji Prediction in English
- Subtask 2: Emoji Prediction in Spanish

TASK DETAILS

Training and Evaluation Data. The data for the task will consist of 500k tweets in English and 100K tweets in Spanish. The tweets were retrieved with the Twitter APIs, from October 2015 to February 2017, and geolocalized in the United States and Spain. The dataset includes tweets that contain one and only one emoji, of the 20 most frequent emojis. Data will be split into trial, training, and test.

Label set. As labels, we will use the 20 most frequent emojis of each language. They are different across the English and Spanish corpora. In the following, we show the distribution of the emojis for each language (numbers refer to the percentage of occurrence of each emoji).

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
USA	❤️	😍	😂	💕	🔥	😊	😎	✨	💙	😜	📷	🇺🇸	☀️	💜	😏	💯	😁	🎄	📺	😜
ESP	❤️	😍	😂	💕	😊	😘	💪	😏	👉	🇪🇸	😎	💙	💜	😜	💕	✨	🎵	💕	😁	👉 TOP

WHAT IS MACHINE LEARNING ?

“ Machine learning is a field of computer science that uses statistical techniques to give computer systems the ability to "learn" with data, without being explicitly programmed”.

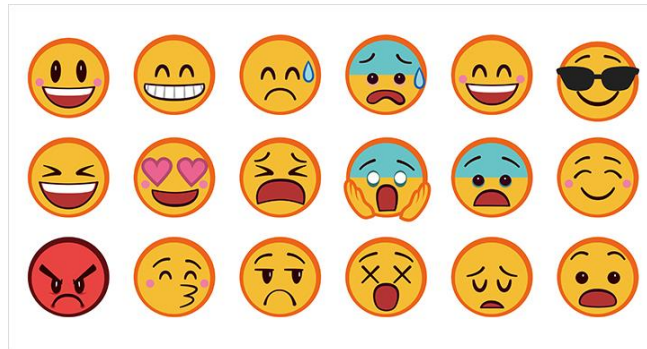
Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. **Machine learning focuses on the development of computer programs** that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. **The primary aim is to allow the computers learn automatically** without human intervention or assistance and adjust actions accordingly.

WHAT ARE EMOJIS ?

“A small digital image or icon used to express an idea or emotion in electronic communication”.

Emoji are ideograms and smileys used in electronic messages and web pages. Emoji exist in various genres, including facial expressions, common objects, places and types of weather, and animals.



OBJECTIVE

- Easy to Interpret the Sentiments of the People posting posts on Social media platform such as Twitter, Facebook etc.
- Easier for Communication Process - While communicating in text messages, we can send the emojis to predict the sentence we want to communicate.
- Better to understand everything - It becomes easy to understand while we are unable to convey something.
- Popping Chatbot Services - Allow for highly engaging, conversational experiences, through voice and text that can be customized and used on mobile devices, web browsers, and on popular chat platforms such as Facebook Messenger, etc.
- Easy Product Review – It becomes easy for customer to Review products with emojis rather than writing the review.
- Easy to Express Feeling – It becomes easy for people to express feeling by emojis.

IMPLEMENTATION DETAILS

There are few Steps involved in Machine Learning :

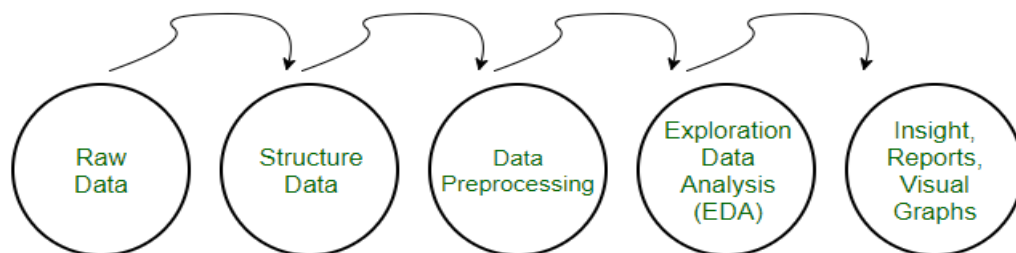
- Gathering Data
- Preprocessing the Data
- Feature Extraction
- Selection of Model
- Training
- Evaluation/Testing

- **Gathering Data** : The tweets were retrieved with the Twitter APIs, from October 2015 to February 2017, and geolocalized in United States and Spain. Tweets contain one and only one emoji.

- No. of tweets in English -500k
- No. of tweets in Spanish – 100k

- **Preprocessing the Data** : Pre-processing refers to the transformations applied to our data before feeding it to the algorithm.

Data Preprocessing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis.



Steps Involved in Pre Processing :

1. **Repeated Characters** - Removing repeated characters in a string.

Example : shoooooooooot -> shoot

2. **Repeated Tweets** - Removing repeated tweet and reduced to a single tweet.

Example :

Jaypee is a Good University -> Jaypee is a good university

Jaypee is a good university

3. **Tokenization** - Tokenization is a step which splits longer strings of text into smaller pieces, or tokens. Larger chunks of text can be tokenized into sentences, sentences can be tokenized into words, etc.

Example : Hello How are you ? -> ['Hello', 'How', 'are', 'you', '?']

4. **Stemming** - “Stemming is the process for reducing inflected words to their word stem (base form).”

Example : Investing, Invested -> Invest

5. **Lower Case** - “Lower Case is the process of converting Upper Case Characters to Lower Case in a String.”

Example : How are you Ram ? -> how are you ram ?

6. **Stop Words** – “These are usually high frequency words that aren’t giving any additional information to our labeling. In English, they could be *the, is, at, which,* and *on*”.

Example : what is on the table ? -> What table ?

Steps Involved in Pre Processing (Continued) :

7. **Parts of Speech** – “Part-of-Speech tagging is the process of reading natural language text and assigning parts of speech to each token.”

Example : The dog ran away. ->

The[*article*] dog[*noun*] ran[*verb*] away[*adjective*].

- **Feature Extraction** : Feature extraction involves reducing the amount of resources required to describe a large set of data. When performing analysis of complex data one of the major problems stems from the number of variables involved. Analysis with a large number of variables generally requires a large amount of memory and computation power, also it may cause a classification algorithm to [overfit](#) to training samples and generalize poorly to new samples. Feature extraction is a general term for methods of constructing combinations of the variables to get around these problems while still describing the data with sufficient accuracy. Many machine learning practitioners believe that properly optimized feature extraction is the key to effective model construction. Feature extraction is related to dimensionality reduction.

Steps Involved in Feature Extraction :

1. **Frequency of Words** - “Calculate the frequency that each word appears in a document out of all the words in the document”.

Example : India 3

Country 5

University 10

Steps Involved in Feature Extraction (Continued) :

2. **Count Special Characters** - “ It is used to count the occurrence of Special Characters”.

Example : ! 100

? 343

3. **Synonyms** – “Synonyms shows the semantic relations among words”.

Example – { ‘beneficial’, ‘just’, ‘upright’, ‘thoroughly’, ‘in_force’, ‘well’, ‘skilful’, ‘skillful’, ‘sound’, ‘unspoiled’, ‘expert’, }

4. **N Grams** - An n-gram is a contiguous sequence of n items from a given sample of text or speech. The items can be phonemes, syllables, letters, words or base pairs according to the application.

They can be ‘Fragmentary strings’, ‘Meaningful strings’, Semantic/Pragmatic expressions.

Examples :

the hell are you

a number of

5. **Uni Gram** - An n-gram of size 1. It can regard words one by one

Examples :

Suppose there is a sentence – World is full of Great People, then its unigram classification will be as follows :-

World

is

full

of

Great

People

6. **Bi Gram** - It is a pair of consecutive written units such as letters, syllables, or words. An n-gram of size 2. It can regard words two at a time.

Examples :

Suppose there is a sentence – World is full of Great People, then its bigram classification will be as follows :-

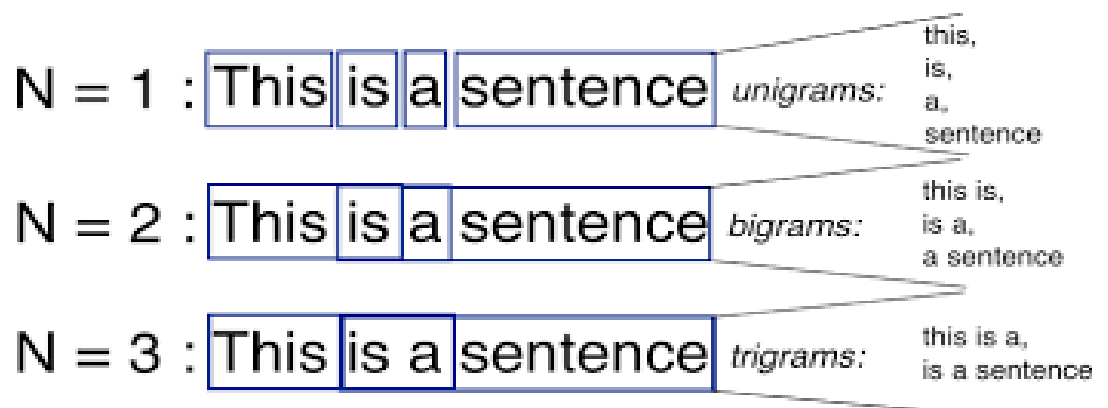
World is
is full
full of
of Great
Great People

7. **Tri Gram** - It is a group of three consecutive written units such as letters, syllables, or words. A n-gram of size 3. It can regard words three at a time

Examples :

Suppose there is a sentence – World is full of Great People, then its trigram classification will be as follows :-

World is full
is full of
full of Great
of Great People



- **Selection of Model** : **Model selection** is the task of selecting a statistical model from a set of candidate models, given data. In the simplest cases, a pre-existing set of data is considered. However, the task can also involve the design of experiments such that the data collected is well-suited to the problem of model selection. Given candidate models of similar predictive or explanatory power, the simplest model is most likely to be the best choice.

1. **NAÏVE BAYES**

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. It is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness, and diameter features.

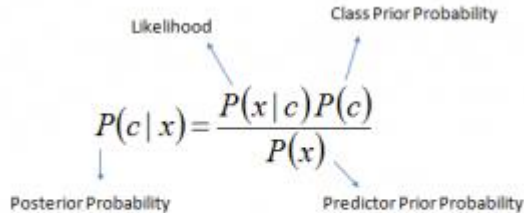
For some types of probability models, naive Bayes classifiers can be trained very efficiently in a supervised learning setting.

It is a classification technique based on [Bayes' Theorem](#) with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as 'Naive'.

Naïve Bayes (Continued) :

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability $P(c|x)$ from $P(c)$, $P(x)$ and $P(x|c)$. Look at the equation below:



The diagram shows the equation $P(c|x) = \frac{P(x|c)P(c)}{P(x)}$ with four labels and arrows: 'Likelihood' points to $P(x|c)$, 'Class Prior Probability' points to $P(c)$, 'Posterior Probability' points to $P(c|x)$, and 'Predictor Prior Probability' points to $P(x)$.

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$
$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \dots \times P(x_n|c) \times P(c)$$

Above,

- $P(c|x)$ is the posterior probability of *class* (c , *target*) given *predictor* (x , *attributes*).
- $P(c)$ is the prior probability of *class*.
- $P(x|c)$ is the likelihood which is the probability of *predictor* given *class*.
- $P(x)$ is the prior probability of *predictor*.

How Naive Bayes algorithm works?

Step 1: Convert the data set into a frequency table

Step 2: Create Likelihood table by finding the probabilities like Overcast probability = 0.29 and probability of playing is 0.64.

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

Selection of Model (Continued) :

2. Stochastic Gradient Descent

Stochastic gradient descent (often shortened to **SGD**), also known as **incremental** gradient descent, is a stochastic approximation of the gradient descent optimization and iterative method for minimizing an objective function that is written as a sum of differentiable functions. In other words, SGD tries to find minima or maxima by iteration. Trained by assigning weights and then updating iteratively until convergence at a maximum.

It is used in Fast Training, Fast Classification and the answer is Sufficiently accurate.

Gradient descent can be slow to run on very large datasets.

Because one iteration of the gradient descent algorithm requires a prediction for each instance in the training dataset, it can take a long time when you have many millions of instances.

In situations when you have large amounts of data, you can use a variation of gradient descent called stochastic gradient descent.

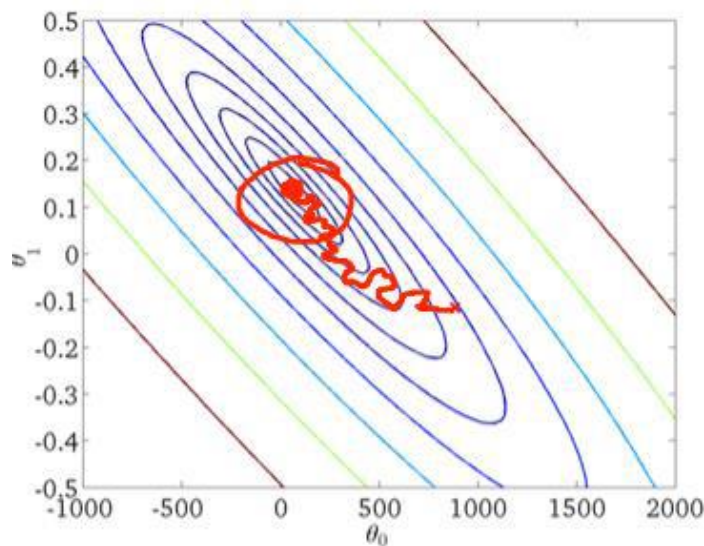
In this variation, the gradient descent procedure described above is run but the update to the coefficients is performed for each training instance, rather than at the end of the batch of instances.

The first step of the procedure requires that the order of the training dataset is randomized. This is to mix up the order that updates are made to the coefficients. Because the coefficients are updated after every training instance, the updates will be noisy jumping all over the place, and so will the corresponding cost function. By mixing up the order for the updates to the coefficients, it harnesses this random walk and avoids it getting distracted or stuck.

The update procedure for the coefficients is the same as that above, except the cost is not summed over all training patterns, but instead calculated for one training pattern.

Selection of Model (Continued) :

The learning can be much faster with stochastic gradient descent for very large training datasets and often you only need a small number of passes through the dataset to reach a good or good enough set of coefficients, e.g. 1-to-10 passes through the dataset.



TRAINING : A training dataset is a dataset of examples used for learning, that is to fit the parameters (e.g., weights) of, for example, a classifier.

Most approaches that search through training data for empirical relationships tend to [overfit](#) the data, meaning that they can identify apparent relationships in the training data that do not hold in general.

Algorithms learn from data. They find relationships, develop understanding, make decisions, and evaluate their confidence from the training data they're given. And the better the training data is, the better the model performs.

In fact, the quality and quantity of your training data has as much to do with the success of your data project as the algorithms themselves.

Training (Continued) :

Data has been trained using different Algorithms and extracting features.

```
Correctly Classified Instances      65          65      %
Incorrectly Classified Instances    35          35      %
Kappa statistic                    0.2994
Mean absolute error                 0.4178
Root mean squared error             0.4578
Relative absolute error             83.7009 %
Root relative squared error         91.6247 %
Total Number of Instances          100

=== Detailed Accuracy By Class ===

          TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Cl
          0.654    0.354    0.667     0.654    0.660      0.299    0.729     0.750     Va
          0.646    0.346    0.633     0.646    0.639      0.299    0.729     0.719     Va
Weighted Avg.    0.650    0.350    0.650     0.650    0.650      0.299    0.729     0.735

=== Confusion Matrix ===

  a  b  <-- classified as
34 18 | a = Value1
17 31 | b = Value2
```

```
Correctly Classified Instances      59          59      %
Incorrectly Classified Instances    41          41      %
Kappa statistic                    0.178
Mean absolute error                 0.4538
Root mean squared error             0.5028
Relative absolute error             90.8295 %
Root relative squared error         100.5749 %
Total Number of Instances          100

=== Detailed Accuracy By Class ===

          TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Cl
          0.615    0.438    0.604     0.615    0.610      0.178    0.624     0.604     Va
          0.563    0.385    0.574     0.563    0.568      0.178    0.624     0.607     Va
Weighted Avg.    0.590    0.412    0.590     0.590    0.590      0.178    0.624     0.605

=== Confusion Matrix ===

  a  b  <-- classified as
32 20 | a = Value1
21 27 | b = Value2
```

- **TESTING** : For Testing , same procedure has been followed as done in Training Part.

The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.

```

Correctly Classified Instances      60          60      %
Incorrectly Classified Instances    40          40      %
Kappa statistic                    0.1961
Mean absolute error                0.4664
Root mean squared error            0.5008
Relative absolute error            93.3564 %
Root relative squared error        100.1666 %
Total Number of Instances         100

```

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Cl
	0.654	0.458	0.607	0.654	0.630	0.197	0.610	0.614	Va
	0.542	0.346	0.591	0.542	0.565	0.197	0.610	0.565	Va
Weighted Avg.	0.600	0.404	0.599	0.600	0.599	0.197	0.610	0.591	

=== Confusion Matrix ===

```

a b  <-- classified as
34 18 | a = Value1
22 26 | b = Value2

```

```

Correctly Classified Instances      52          52      %
Incorrectly Classified Instances     48          48      %
Kappa statistic                    0
Mean absolute error                0.4992
Root mean squared error            0.4996
Relative absolute error            100      %
Root relative squared error        100      %
Total Number of Instances         100

```

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Cl
	1.000	1.000	0.520	1.000	0.684	?	0.500	0.520	Va
	0.000	0.000	?	0.000	?	?	0.500	0.480	Va
Weighted Avg.	0.520	0.520	?	0.520	?	?	0.500	0.501	

=== Confusion Matrix ===

```

a b  <-- classified as
52  0 | a = Value1
48  0 | b = Value2

```


CODES

LEMMA :

```
from nltk.stem import WordNetLemmatizer

from nltk.tokenize import word_tokenize
fh=open("data/Sentence.txt","r")
rq=fh.read()
f=open('data/lemma.txt','w')
for i in word_tokenize(rq):
    lemmatizer = WordNetLemmatizer()

    f.write(lemmatizer.lemmatize(i))
    f.write('\n')
f.close
```

PART OF SPEECH :

```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

stop_words = set(stopwords.words('english'))

fh=open("data/Sentence.txt","r")
txt=fh.read()

# sent_tokenize is one of instances of
# PunktSentenceTokenizer from the nltk.tokenize.punkt module
f=open("data/pos.txt","w")
tokenized = word_tokenize(txt)
for i in tokenized:
    wordsList = nltk.word_tokenize(i)
    wordsList = [w for w in wordsList if not w in stop_words]
    tagged = nltk.pos_tag(wordsList)
    f.write(str(tagged))
    f.write('\n')
f.close()
```

LOWERCASE :

```
from itertools import chain
from glob import glob

file = open('data/Sentence.txt', 'r')

lines = [line.lower() for line in file]
with open('data/lower.txt', 'w') as out:
    out.writelines((lines))
```

STEMMING :

```
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
ps=PorterStemmer()

fh=open("data/Sentence.txt","r")
rq=fh.read()

f=open('data/stem.txt','w')

words=word_tokenize(rq)

for w in words:
    f.write(ps.stem(w))
    f.write('\n')
f.close
```

STOP WORDS :

```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

stop_words=set(stopwords.words("english"))
fh=open("unigram.text","r")
rq=fh.read()

words=word_tokenize(rq)
filtered_sentence=[]
f=open('stop1.txt','w')
for w in words:
    if w not in stop_words:
        f.write(w)
        f.write('\n')
f.close
```

TOKENIZE :

```
from nltk.tokenize import word_tokenize
fh=open("Dataset.txt","r")
rq=fh.read()
f=open('tokenize.txt','w')
for i in word_tokenize(rq):
    f.write(i)
    f.write('\n')
f.close
```

OUTPUT

Getting Twitter API Keys :

1. Getting Twitter API keys

To start with, you will need to have a Twitter account and obtain credentials (i.e. API key, API secret, Access token and Access token secret) on the Twitter developer site to access the Twitter API, following these steps:

- Create a Twitter user account if you do not already have one.
- Go to <https://apps.twitter.com/> and log in with your Twitter user account. This step gives you a Twitter dev account under the same name as your user account.
- Click "Create New App"
- Fill out the form, agree to the terms, and click "Create your Twitter application"
- In the next page, click on "Keys and Access Tokens" tab, and copy your "API key" and "API secret". Scroll down and click "Create my access token", and copy your "Access token" and "Access token secret".

2. Installing a Twitter library

We will be using a Python library called [Python Twitter Tools](#) to connect to Twitter API and downloading the data from Twitter. There are [many other libraries](#) in various programming languages that let you use Twitter API. We choose the Python Twitter Tools for this tutorial, because it is simple to use yet fully supports the Twitter API.

Download the Python Twitter tools at <https://pypi.python.org/pypi/twitter>.

Install the Python Twitter Tools package by typing in commands:

```
$ python setup.py --help
$ python setup.py build
$ python setup.py install
```

3. Connecting to Twitter Streaming APIs

The Streaming APIs give access to (usually a sample of) all tweets as they published on Twitter. On average, about 6,000 tweets per second are posted on Twitter and you (normal dev users) will get a small proportion ($\leq 1\%$) of it. The Streaming APIs are one of the two types of Twitter APIs. The other one called REST APIs (we will talk about later in this tutorial), which is more suitable for singular searches, such as searching historic tweets, reading user profile information, or posting Tweets. The Streaming API **only** sends out real-time tweets, while the Search API (one of the popular REST APIs) gives historical tweets up to about a week with a max of a couple of hundreds. You may request elevated access (e.g. Firehose, Retweet, Link, Birdog or Shadow) for more data by contacting Twitter's API support.

Basic Uses of Streaming APIs

Create a file called `twitter_streaming.py`, and copy the code below into it. Make sure to enter your credentials obtained in the Step 1 above into `ACCESS_TOKEN`, `ACCESS_SECRET`, `CONSUMER_KEY`, and `CONSUMER_SECRET`.

Application Settings

Your application's Consumer Key and Secret are used to [authenticate requests to the Twitter Platform](#).

Access level	Read and write (modify app permissions)
Consumer Key (API Key)	TgRlnrxyJvhZjbuDF55qSc15 (manage keys and access tokens)
Callback URL	None
Callback URL Locked	No
Sign in with Twitter	Yes
App-only authentication	https://api.twitter.com/oauth2/token
Request token URL	https://api.twitter.com/oauth/request_token
Authorize URL	https://api.twitter.com/oauth/authorize
Access token URL	https://api.twitter.com/oauth/access_token

Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key) TgRlnrxyJvhZjbuDF55qSc15

Consumer Secret (API Secret) VCX4WLry4td2SJ8di63SkZQizGX9nqVjRdM6M0JHRI0tQhYKUx

Access Level Read and write (modify app permissions)

Owner ebullient_mind

Owner ID 936189948339789825

TWEETS IN ENGLISH DATASETS :

LoL @ West Covina, California
Things got a little festive at the office #christmas2016 @
RedRockâ€!
Step out and explore. # i, @ Ellis Island Cafe
@user @ Cathedral Preparatory School
My baby bear @ Bubby's
RuPaul's Drag Race bingo fun. Drag Queens be SEXY!
#rupaulsdragrace @user abwyman #laâ€!
Black History like a Mufffffaaaaaka #blacchyna done thru her
yugioh trap card like hell â€!
Just light makeup i, #blueeyes #lupusgirl #photography
#modelingagency #modeling #smilingâ€!
@ BJ's Restaurant and Brewhouse
So lovely catching up with my soul sister @user @ University of
Victoria
Perfect for this weather i, #dessert #snowice #snowwhite
#lasvegas #summer @ Snow White Cafe
Had fun (at @user in New York, NY)
Well Damn @ Oklahoma City, Oklahoma
'scuse me while I kiss the sky. ____ : nikkileekv @ Malibu,
California
Fun in the sun i, @ Brownstone Park, Portland, CT
Celebrating #LAsyle @ Calle Tacos
I think today is about to be a great day..
@user it's all Bama now What's your take on Kiffin? Ask a #Bama
fan.
Bae-rito Wednesday. the day we convinced the girl that's
addictedâ€!
Birthday Ski Trip with ny sweet hubby! #OneYearAwayFrom30

TWEETS IN SPANISH DATASETS :

Plaza de Oriente , Madrid#madrid #city #plazadeorient #puertadesol #tour...↓
Por ser la columna de mi templo, por ser lo mejor que tengo. @ Parquesur↓
Me gustan las motos! #cheste2016 #nicoabad #elañoquevienemás @user↓
Sevilla tiene un color especial, Sevilla tiene un color diferente. #Sevilla #SemanaSanta...↓
Que (la) Chipi no se caiga .Cuánto os quiero chavales!! @ Valdemoro Centro↓
Haciendo el tonto la vida se vive mucho mejor @ Pantano San Juan, Pelayos De La Presa.↓
DANI MARTÍN Más de 7 horas de cola merecieron la pena!!...↓
Tras una semana de locura, se acaba el Arenal Sound pero nos quedan recuerdos inolvidables! ...↓
Todo más que dicho anilota, disfruta mucho ya queda menos para los...↓
Cerrando el Ayuntamiento (@ Ayuntamiento de San Sebastián de los Reyes in San Sebastián de los Reyes, Madrid)↓
Es imposible estudiar cuando alguien te peta mandando 37 audios en serio @user para ya pesa'↓
"Quiero ser tuya enterita pero tengo miedo." @ Parla↓
Perfecto. Te quiero @ Pabellón Municipal De Talayuela↓
Miss my flawless fabulososssss @ Primark Gran Vía Madrid↓
Cada día más humana, menos perfecta y más feliz. @ Albalá, Spain↓
Isi con 2 patitos Te quiero mucho osita y espero que lo...↓
El ceviche de Ancón↓
Cuando vuelves andando de Ceutí #ánimo @ En La Mierda↓
Todo esto para mi #playa #sol #arena #rocas #formentera #estaes_baleares #verano2016...↓
Lamento que no estés cuando amanece. @ La Bañeza, Spain↓
Si no sabes explicarlo es porque merece la pena @ Toledo, Spain↓
#LQQCQ #loquequierascuandoquieras y con quienes quieras #friends...↓
Muchísimas felicidades guapisimas, por muchos mas con vosotras ...↓
BUSCANDO EN EL BAÚL DE LOS RECUERDOS, UUUU(8) @ Mérida, Spain↓
De comida con la family a celebrar los 2 patitos de la enana.. @user↓