https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/

* [**Start Here**](javascript:void(0);)
* [Articles](javascript:void(0);)
* [Courses](javascript:void(0);)
* [Job-A-Thon](https://datahack.analyticsvidhya.com/contest/job-a-thon-september-2021/?utm_source=blog_navbar&utm_medium=hackathon_button)
* **A**

Get a Guaranteed\* Job with Data Science Immersive Bootcamp

[Download Brochure](https://bootcamp.analyticsvidhya.com/?utm_source=blog&utm_medium=flashstrip)

×

1. [Home](https://www.analyticsvidhya.com/)

**Comprehensive Hands on Guide to Twitter Sentiment Analysis with dataset and code**

[**Prateek Joshi**](https://www.analyticsvidhya.com/blog/author/pjoshi15/)**— July 30, 2018**

[Advanced](https://www.analyticsvidhya.com/blog/category/advanced/) [Classification](https://www.analyticsvidhya.com/blog/category/classification/) [Machine Learning](https://www.analyticsvidhya.com/blog/category/machine-learning/) [NLP](https://www.analyticsvidhya.com/blog/category/nlp/) [Project](https://www.analyticsvidhya.com/blog/category/project/) [Python](https://www.analyticsvidhya.com/blog/category/python-2/) [Supervised](https://www.analyticsvidhya.com/blog/category/supervised/) [Text](https://www.analyticsvidhya.com/blog/category/text/) [Unstructured Data](https://www.analyticsvidhya.com/blog/category/unstructured-data/)

Introduction

Natural Language Processing (NLP) is a hotbed of research in data science these days and one of the most common applications of NLP is sentiment analysis. From opinion polls to creating entire marketing strategies, this domain has completely reshaped the way businesses work, which is why this is an area every data scientist must be familiar with.

Thousands of text documents can be processed for sentiment (and other features including named entities, topics, themes, etc.) in seconds, compared to the hours it would take a team of people to manually complete the same task.



In this article, we will learn how to solve the [Twitter Sentiment Analysis Practice Problem](https://datahack.analyticsvidhya.com/contest/practice-problem-twitter-sentiment-analysis/).

We will do so by following a sequence of steps needed to solve a general sentiment analysis problem. We will start with preprocessing and cleaning of the raw text of the tweets. Then we will explore the cleaned text and try to get some intuition about the context of the tweets. After that, we will extract numerical features from the data and finally use these feature sets to train models and identify the sentiments of the tweets.

This is one of the most interesting challenges in NLP so I’m very excited to take this journey with you!

Table of Contents

1. Understand the Problem Statement
2. Tweets Preprocessing and Cleaning
3. Story Generation and Visualization from Tweets
4. Extracting Features from Cleaned Tweets
5. Model Building: Sentiment Analysis
6. What’s Next

1. Understand the Problem Statement

Let’s go through the problem statement once as it is very crucial to understand the objective before working on the dataset. The problem statement is as follows:

***The objective of this task is to detect hate speech in tweets. For the sake of simplicity, we say a tweet contains hate speech if it has a racist or sexist sentiment associated with it. So, the task is to classify racist or sexist tweets from other tweets.***

Formally, given a training sample of tweets and labels, where label ‘1’ denotes the tweet is racist/sexist and label ‘0’ denotes the tweet is not racist/sexist, your objective is to predict the labels on the given test dataset.

*Note: The evaluation metric from this practice problem is****F1-Score****.*

Personally, I quite like this task because hate speech, trolling and social media bullying have become serious issues these days and a system that is able to detect such texts would surely be of great use in making the internet and social media a better and bully-free place. Let’s look at each step in detail now.

2. Tweets Preprocessing and Cleaning

Take a look at the pictures below depicting two scenarios of an office space – one is untidy and the other is clean and organized.



You are searching for a document in this office space. In which scenario are you more likely to find the document easily? Of course, in the less cluttered one because each item is kept in its proper place. The data cleaning exercise is quite similar. If the data is arranged in a structured format then it becomes easier to find the right information.

The preprocessing of the text data is an essential step as it makes the raw text ready for mining, i.e., it becomes easier to extract information from the text and apply machine learning algorithms to it. If we skip this step then there is a higher chance that you are working with noisy and inconsistent data. The objective of this step is to clean noise those are less relevant to find the sentiment of tweets such as punctuation, special characters, numbers, and terms which don’t carry much weightage in context to the text.

In one of the later stages, we will be extracting numeric features from our Twitter text data. This feature space is created using all the unique words present in the entire data. So, if we preprocess our data well, then we would be able to get a better quality feature space.

Let’s first read our data and load the necessary libraries. You can download the datasets from [here](https://datahack.analyticsvidhya.com/contest/practice-problem-twitter-sentiment-analysis/).

**import** re

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

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

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

**import** seaborn **as** sns

**import** string

**import** nltk

**import** warnings

warnings.filterwarnings("ignore", category=DeprecationWarning)

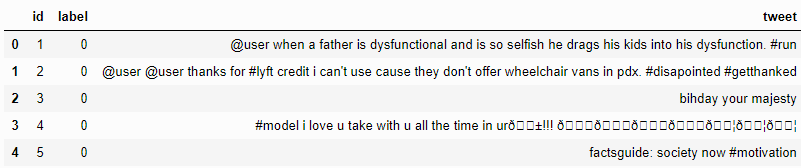
%matplotlib inline

train  = pd.read\_csv('train\_E6oV3lV.csv')

test = pd.read\_csv('test\_tweets\_anuFYb8.csv')

Let’s check the first few rows of the train dataset.

train.head()



The data has 3 columns ***id****,****label***, and ***tweet***. ***label*** is the binary target variable and ***tweet***contains the tweets that we will clean and preprocess.

Initial data cleaning requirements that we can think of after looking at the top 5 records:

* The Twitter handles are already masked as @user due to privacy concerns. So, these Twitter handles are hardly giving any information about the nature of the tweet.
* We can also think of getting rid of the punctuations, numbers and even special characters since they wouldn’t help in differentiating different kinds of tweets.
* Most of the smaller words do not add much value. For example, ‘pdx’, ‘his’, ‘all’. So, we will try to remove them as well from our data.
* Once we have executed the above three steps, we can split every tweet into individual words or tokens which is an essential step in any NLP task.
* In the 4th tweet, there is a word ‘love’. We might also have terms like loves, loving, lovable, etc. in the rest of the data. These terms are often used in the same context. If we can reduce them to their root word, which is ‘love’, then we can reduce the total number of unique words in our data without losing a significant amount of information.

A) Removing Twitter Handles (@user)

As mentioned above, the tweets contain lots of twitter handles (@user), that is how a Twitter user acknowledged on Twitter. We will remove all these twitter handles from the data as they don’t convey much information.

For our convenience, let’s first combine train and test set. This saves the trouble of performing the same steps twice on test and train.

combi = train.append(test, ignore\_index=**True**)

Given below is a user-defined function to remove unwanted text patterns from the tweets. It takes two arguments, one is the original string of text and the other is the pattern of text that we want to remove from the string. The function returns the same input string but without the given pattern. We will use this function to remove the pattern ‘@user’ from all the tweets in our data.

**def** **remove\_pattern**(input\_txt, pattern):

   r = re.findall(pattern, input\_txt)

**for** i **in** r:

       input\_txt = re.sub(i, '', input\_txt)

**return** input\_txt

Now let’s create a new column **tidy\_tweet,**it will contain the cleaned and processed tweets. Note that we have passed “@[\w]\*” as the pattern to the *remove\_pattern* function. It is actually a regular expression which will pick any word starting with ‘@’.

*# remove twitter handles (@user)*

combi['tidy\_tweet'] = np.vectorize(remove\_pattern)(combi['tweet'], "@[\w]\*")

B) Removing Punctuations, Numbers, and Special Characters

As discussed, punctuations, numbers and special characters do not help much. It is better to remove them from the text just as we removed the twitter handles. Here we will replace everything except characters and hashtags with spaces.

*# remove special characters, numbers, punctuations*

combi['tidy\_tweet'] = combi['tidy\_tweet'].str.replace("[^a-zA-Z#]", " ")

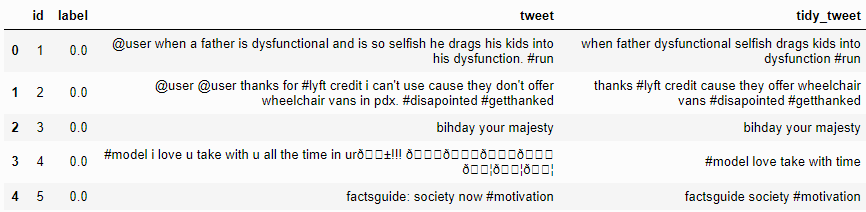
C) Removing Short Words

We have to be a little careful here in selecting the length of the words which we want to remove. So, I have decided to remove all the words having length 3 or less. For example, terms like “hmm”, “oh” are of very little use. It is better to get rid of them.

combi['tidy\_tweet'] = combi['tidy\_tweet'].apply(**lambda** x: ' '.join([w **for** w **in** x.split() **if** len(w)>3]))

Let’s take another look at the first few rows of the combined dataframe.

combi.head()



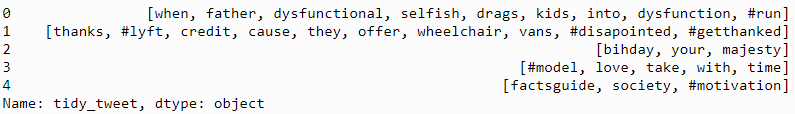
You can see the difference between the raw tweets and the cleaned tweets (tidy\_tweet) quite clearly. Only the important words in the tweets have been retained and the noise (numbers, punctuations, and special characters) has been removed.

D) Tokenization

Now we will tokenize all the cleaned tweets in our dataset. Tokens are individual terms or words, and tokenization is the process of splitting a string of text into tokens.

tokenized\_tweet = combi['tidy\_tweet'].apply(**lambda** x: x.split())

tokenized\_tweet.head()



E) Stemming

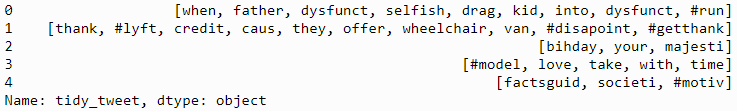
Stemming is a rule-based process of stripping the suffixes (“ing”, “ly”, “es”, “s” etc) from a word. For example, For example – “play”, “player”, “played”, “plays” and “playing” are the different variations of the word – “play”.

from nltk.stem.porter import \*

stemmer = PorterStemmer()

tokenized\_tweet = tokenized\_tweet.apply(lambda x: [stemmer.stem(i) for i in x]) # stemming

tokenized\_tweet.head()



Now let’s stitch these tokens back together.

for i in range(len(tokenized\_tweet)):

tokenized\_tweet[i] = ' '.join(tokenized\_tweet[i])

combi['tidy\_tweet'] = tokenized\_tweet

3. Story Generation and Visualization from Tweets

In this section, we will explore the cleaned tweets text. Exploring and visualizing data, no matter whether its text or any other data, is an essential step in gaining insights. Do not limit yourself to only these methods told in this tutorial, feel free to explore the data as much as possible.

Before we begin exploration, we must think and ask questions related to the data in hand. A few probable questions are as follows:

* What are the most common words in the entire dataset?
* What are the most common words in the dataset for negative and positive tweets, respectively?
* How many hashtags are there in a tweet?
* Which trends are associated with my dataset?
* Which trends are associated with either of the sentiments? Are they compatible with the sentiments?

A) Understanding the common words used in the tweets: WordCloud

Now I want to see how well the given sentiments are distributed across the train dataset. One way to accomplish this task is by understanding the common words by plotting wordclouds.

*A wordcloud is a visualization wherein the most frequent words appear in large size and the less frequent words appear in smaller sizes.*

Let’s visualize all the words our data using the wordcloud plot.

all\_words = ' '.join([text **for** text **in** combi['tidy\_tweet']])

from wordcloud **import** WordCloud

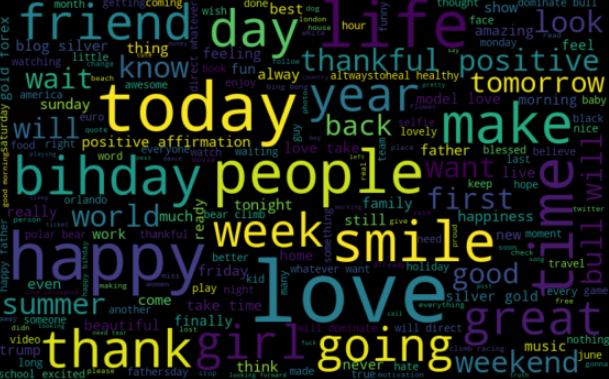
wordcloud = WordCloud(width=800, height=500, random\_state=21, max\_font\_size=110).generate(all\_words)

plt.figure(figsize=(10, 7))

plt.imshow(wordcloud, interpolation="bilinear")

plt.axis('off')

plt.show()



We can see most of the words are positive or neutral. With *happy* and *love* being the most frequent ones. It doesn’t give us any idea about the words associated with the racist/sexist tweets. Hence, we will plot separate wordclouds for both the classes(racist/sexist or not) in our train data.

**B) Words in non racist/sexist tweets**

normal\_words =' '.join([text **for** text **in** combi['tidy\_tweet'][combi['label'] == 0]])

wordcloud = WordCloud(width=800, height=500, random\_state=21, max\_font\_size=110).generate(normal\_words)

plt.figure(figsize=(10, 7))

plt.imshow(wordcloud, interpolation="bilinear")

plt.axis('off')

plt.show()



We can see most of the words are positive or neutral. With *happy, smile,*and *love* being the most frequent ones. Hence, most of the frequent words are compatible with the sentiment which is non racist/sexists tweets. Similarly, we will plot the word cloud for the other sentiment. Expect to see negative, racist, and sexist terms.

**C) Racist/Sexist Tweets**

negative\_words = ' '.join([text for text in combi['tidy\_tweet'][combi['label'] == 1]])

wordcloud = WordCloud(width=800, height=500,

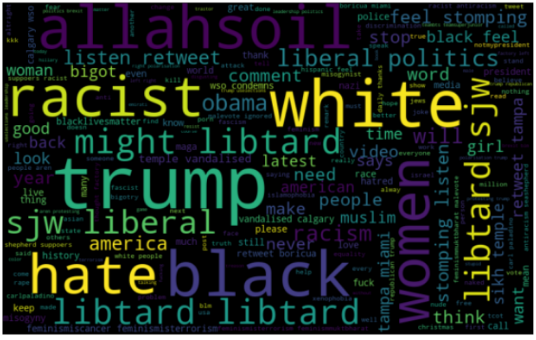
random\_state=21, max\_font\_size=110).generate(negative\_words)

plt.figure(figsize=(10, 7))

plt.imshow(wordcloud, interpolation="bilinear")

plt.axis('off')

plt.show()



As we can clearly see, most of the words have negative connotations. So, it seems we have a pretty good text data to work on. Next we will the hashtags/trends in our twitter data.

D) Understanding the impact of Hashtags on tweets sentiment

Hashtags in twitter are synonymous with the ongoing trends on twitter at any particular point in time. We should try to check whether these hashtags add any value to our sentiment analysis task, i.e., they help in distinguishing tweets into the different sentiments.

For instance, given below is a tweet from our dataset:



The tweet seems sexist in nature and the hashtags in the tweet convey the same feeling.

We will store all the trend terms in two separate lists — one for non-racist/sexist tweets and the other for racist/sexist tweets.

*# function to collect hashtags*

**def** **hashtag\_extract**(x):

   hashtags = []

*# Loop over the words in the tweet*

**for** i **in** x:

       ht = re.findall(r"#(\w+)", i)

       hashtags.append(ht)

**return** hashtags

*# extracting hashtags from non racist/sexist tweets*

HT\_regular = hashtag\_extract(combi['tidy\_tweet'][combi['label'] == 0])

*# extracting hashtags from racist/sexist tweets*

HT\_negative = hashtag\_extract(combi['tidy\_tweet'][combi['label'] == 1])

*# unnesting list*

HT\_regular = sum(HT\_regular,[])

HT\_negative = sum(HT\_negative,[])

Now that we have prepared our lists of hashtags for both the sentiments, we can plot the top n hashtags. So, first let’s check the hashtags in the non-racist/sexist tweets.

**Non-Racist/Sexist Tweets**

a = nltk.FreqDist(HT\_regular)

d = pd.DataFrame({'Hashtag': list(a.keys()),

                 'Count': list(a.values())})

*# selecting top 10 most frequent hashtags*

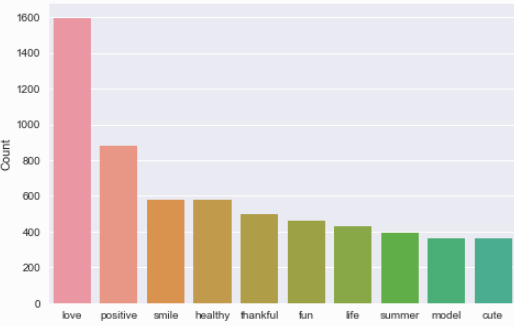
d = d.nlargest(columns="Count", n = 10)

plt.figure(figsize=(16,5))

ax = sns.barplot(data=d, x= "Hashtag", y = "Count")

ax.set(ylabel = 'Count')

plt.show()



All these hashtags are positive and it makes sense. I am expecting negative terms in the plot of the second list. Let’s check the most frequent hashtags appearing in the racist/sexist tweets.

**Racist/Sexist Tweets**

b = nltk.FreqDist(HT\_negative)

e = pd.DataFrame({'Hashtag': list(b.keys()), 'Count': list(b.values())})

*# selecting top 10 most frequent hashtags*

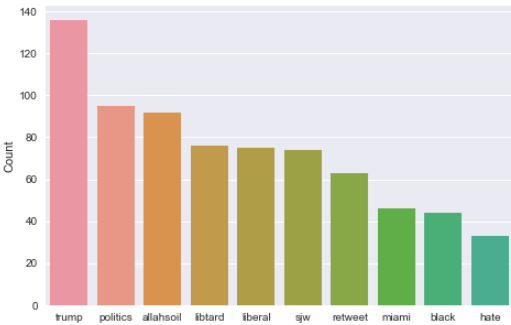
e = e.nlargest(columns="Count", n = 10)

plt.figure(figsize=(16,5))

ax = sns.barplot(data=e, x= "Hashtag", y = "Count")

ax.set(ylabel = 'Count')

plt.show()



As expected, most of the terms are negative with a few neutral terms as well. So, it’s not a bad idea to keep these hashtags in our data as they contain useful information. Next, we will try to extract features from the tokenized tweets.

4. Extracting Features from Cleaned Tweets

To analyze a preprocessed data, it needs to be converted into features. Depending upon the usage, text features can be constructed using assorted techniques – Bag-of-Words, TF-IDF, and Word Embeddings. In this article, we will be covering only Bag-of-Words and TF-IDF.

Bag-of-Words Features

Bag-of-Words is a method to represent text into numerical features. Consider a corpus (a collection of texts) called C of D documents {d1,d2…..dD} and N unique tokens extracted out of the corpus C. The N tokens (words) will form a list, and the size of the bag-of-words matrix M will be given by D X N. Each row in the matrix M contains the frequency of tokens in document D(i).

Let us understand this using a simple example. Suppose we have only 2 document

D1: He is a lazy boy. She is also lazy.

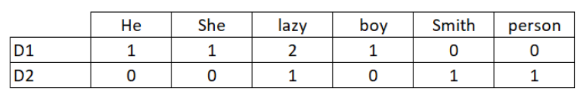
D2: Smith is a lazy person.

The list created would consist of all the unique tokens in the corpus C.

= [‘He’,’She’,’lazy’,’boy’,’Smith’,’person’]

Here, D=2, N=6

The matrix M of size 2 X 6 will be represented as –



Now the columns in the above matrix can be used as features to build a classification model. Bag-of-Words features can be easily created using sklearn’s *CountVectorizer* function. We will set the parameter max\_features = 1000 to select only top 1000 terms ordered by term frequency across the corpus.

**from** sklearn.feature\_extraction.text **import** CountVectorizer

bow\_vectorizer = CountVectorizer(max\_df=0.90, min\_df=2, max\_features=1000, stop\_words='english')

*# bag-of-words feature matrix*

bow = bow\_vectorizer.fit\_transform(combi['tidy\_tweet'])

TF-IDF Features

This is another method which is based on the frequency method but it is different to the bag-of-words approach in the sense that it takes into account, not just the occurrence of a word in a single document (or tweet) but in the entire corpus.

TF-IDF works by penalizing the common words by assigning them lower weights while giving importance to words which are rare in the entire corpus but appear in good numbers in few documents.

Let’s have a look at the important terms related to TF-IDF:

* TF = (Number of times term t appears in a document)/(Number of terms in the document)
* IDF = log(N/n), where, N is the number of documents and n is the number of documents a term t has appeared in.
* TF-IDF = TF\*IDF

**from** sklearn.feature\_extraction.text **import** TfidfVectorizer

tfidf\_vectorizer = TfidfVectorizer(max\_df=0.90, min\_df=2, max\_features=1000, stop\_words='english')

*# TF-IDF feature matrix*

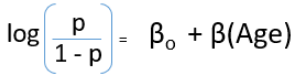
tfidf = tfidf\_vectorizer.fit\_transform(combi['tidy\_tweet'])

5. Model Building: Sentiment Analysis

We are now done with all the pre-modeling stages required to get the data in the proper form and shape. Now we will be building predictive models on the dataset using the two feature set — Bag-of-Words and TF-IDF.

We will use logistic regression to build the models. It predicts the probability of occurrence of an event by fitting data to a logit function.

The following equation is used in Logistic Regression:



Read this [article](https://www.analyticsvidhya.com/blog/2015/11/beginners-guide-on-logistic-regression-in-r/) to know more about Logistic Regression.

*Note: If you are interested in trying out other machine learning algorithms like RandomForest, Support Vector Machine, or XGBoost, then we have a****free full-fledged***[***course on Sentiment Analysis***](https://trainings.analyticsvidhya.com/courses/course-v1:AnalyticsVidhya+TSA001+2018_T1/info)*for you.*

A) Building model using Bag-of-Words features

**from** sklearn.linear\_model **import** LogisticRegression

**from** sklearn.model\_selection **import** train\_test\_split

**from** sklearn.metrics **import** f1\_score

train\_bow = bow[:31962,:]

test\_bow = bow[31962:,:]

*# splitting data into training and validation set*

xtrain\_bow, xvalid\_bow, ytrain, yvalid = train\_test\_split(train\_bow, train['label'], random\_state=42, test\_size=0.3)

lreg = LogisticRegression()

lreg.fit(xtrain\_bow, ytrain) *# training the model*

prediction = lreg.predict\_proba(xvalid\_bow) *# predicting on the validation set*

prediction\_int = prediction[:,1] >= 0.3 *# if prediction is greater than or equal to 0.3 than 1 else 0*

prediction\_int = prediction\_int.astype(np.int)

f1\_score(yvalid, prediction\_int) *# calculating f1 score*

**Output: 0.53**

We trained the logistic regression model on the Bag-of-Words features and it gave us an F1-score of 0.53 for the validation set. Now we will use this model to predict for the test data.

test\_pred = lreg.predict\_proba(test\_bow)

test\_pred\_int = test\_pred[:,1] >= 0.3

test\_pred\_int = test\_pred\_int.astype(np.int)

test['label'] = test\_pred\_int

submission = test[['id','label']]

submission.to\_csv('sub\_lreg\_bow.csv', index=**False**) *# writing data to a CSV file*

The public leaderboard F1 score is 0.567. Now we will again train a logistic regression model but this time on the TF-IDF features. Let’s see how it performs.

B) Building model using TF-IDF features

train\_tfidf = tfidf[:31962,:]

test\_tfidf = tfidf[31962:,:]

xtrain\_tfidf = train\_tfidf[ytrain.index]

xvalid\_tfidf = train\_tfidf[yvalid.index]

lreg.fit(xtrain\_tfidf, ytrain)

prediction = lreg.predict\_proba(xvalid\_tfidf)

prediction\_int = prediction[:,1] >= 0.3

prediction\_int = prediction\_int.astype(np.int)

f1\_score(yvalid, prediction\_int)

**Output: 0.544**

The validation score is 0.544 and the public leaderboard F1 score is 0.564. So, by using the TF-IDF features, the validation score has improved and the public leaderboard score is more or less the same.

6. What’s Next?

If you are interested to learn about more techniques for Sentiment Analysis, we have a well laid out [**video course on NLP**](https://trainings.analyticsvidhya.com/courses/course-v1:AnalyticsVidhya+NLP101+2018_T1/about?utm_source=blog&utm_medium=hands-on-sentiment-analysis-dataset-python/) for you.This course is designed for people who are looking to get into the field of Natural Language Processing. It provides you everything you need to know to become an NLP practitioner.

Key topics covered in the course:

* Extracting named entities from the text
* Topic Modelling
* Feature engineering for text
* Text classification
* Deep Learning for NLP
* 3 real life projects

End Notes

In this article, we learned how to approach a sentiment analysis problem. We started with preprocessing and exploration of data. Then we extracted features from the cleaned text using Bag-of-Words and TF-IDF. Finally, we were able to build a couple of models using both the feature sets to classify the tweets.

Did you find this article useful? Do you have any useful trick? Did you use any other method for feature extraction? Feel free to discuss your experiences in comments below or on the [discussion portal](https://discuss.analyticsvidhya.com/) and we’ll be more than happy to discuss.

Full Code: <https://github.com/prateekjoshi565/twitter_sentiment_analysis/blob/master/code_sentiment_analysis.ipynb>

**Related**

[](https://www.analyticsvidhya.com/blog/2021/06/how-to-perform-basic-text-analysis-without-training-dataset/?relatedposts_hit=1&relatedposts_origin=45979&relatedposts_position=0)

[How to Perform Basic Text Analysis without Training Dataset](https://www.analyticsvidhya.com/blog/2021/06/how-to-perform-basic-text-analysis-without-training-dataset/?relatedposts_hit=1&relatedposts_origin=45979&relatedposts_position=0)

[](https://www.analyticsvidhya.com/blog/2021/06/rule-based-sentiment-analysis-in-python/?relatedposts_hit=1&relatedposts_origin=45979&relatedposts_position=1)

[Rule-Based Sentiment Analysis in Python](https://www.analyticsvidhya.com/blog/2021/06/rule-based-sentiment-analysis-in-python/?relatedposts_hit=1&relatedposts_origin=45979&relatedposts_position=1)

[](https://www.analyticsvidhya.com/blog/2021/07/performing-sentiment-analysis-using-twitter-data/?relatedposts_hit=1&relatedposts_origin=45979&relatedposts_position=2)

[Performing Sentiment Analysis Using Twitter Data!](https://www.analyticsvidhya.com/blog/2021/07/performing-sentiment-analysis-using-twitter-data/?relatedposts_hit=1&relatedposts_origin=45979&relatedposts_position=2)

[live coding](https://www.analyticsvidhya.com/blog/tag/live-coding/)[machine learning](https://www.analyticsvidhya.com/blog/tag/machine-learning/)[Natural language processing](https://www.analyticsvidhya.com/blog/tag/natural-language-processing/)[NLP](https://www.analyticsvidhya.com/blog/tag/nlp/)[python](https://www.analyticsvidhya.com/blog/tag/python/)[sentiment analysis](https://www.analyticsvidhya.com/blog/tag/sentiment-analysis/)[tfidf](https://www.analyticsvidhya.com/blog/tag/tfidf/)[Twitter sentiment analysis](https://www.analyticsvidhya.com/blog/tag/twitter-sentiment-analysis/)

**About the Author**



[Prateek Joshi](https://www.analyticsvidhya.com/blog/author/pjoshi15/)

Data Scientist at Analytics Vidhya with multidisciplinary academic background. Experienced in machine learning, NLP, graphs & networks. Passionate about learning and applying data science to solve real world problems.

**Our Top Authors**

* 
* 
* 
* 
* 
* 
* 

**DownloadAnalytics VidhyaAppfor the Latest blog/Article**

**Previous Post**[An AI Algorithm Detects your Personality by analyzing Eye Movement!](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/)

**Next Post**[Become a Computer Vision Artist with Stanford’s Game Changing ‘Outpainting’ Algorithm (with GitHub link)](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/)

58 thoughts on "Comprehensive Hands on Guide to Twitter Sentiment Analysis with dataset and code"



tom mcgrory says:**July 31, 2018 at 2:44 am**

Thanks you for your work on the twitter sentiment in the article is, there any way to get the article in PDF format? I am new to NLTP / NLTK and would like to work through the article as I look at my own dataset but it is difficult scrolling back and forth as I work.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Nicholas Kemp says:**July 31, 2018 at 7:49 am**

Hello I can't seem to find the data[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**July 31, 2018 at 10:41 am**

Hi Nicholas, You can download the datasets from [here](https://datahack.analyticsvidhya.com/contest/practice-problem-twitter-sentiment-analysis/).[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Ateeque Shaikh says:**July 31, 2018 at 11:54 am**

Can you post R code as well[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



T G Harsha Vardhan says:**August 03, 2018 at 9:52 am**

Hi,Good article.How the raw tweets are given a sentiment(Target variable) and made it into a supervised learning.Is it done by polarity algorithms(text blob)??..In twitter analysis,how the target variable(sentiment) is mapped to incoming tweet is more crucial than classification. Isn't it??[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**August 03, 2018 at 11:40 am**

Thanks for appreciating. The raw tweets were labeled manually.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



NAGA PRUDHVI says:**August 03, 2018 at 7:21 pm**

Hi this was good explination. But how can our model or system knows which are happy words and which are racist/sexist words.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Lilya says:**August 06, 2018 at 7:44 am**

Hi, Thank you for your kind information, but I have one question that in this part, you just analyze the sentiment of single rather than the whole sentence, so some bad circumstance may happen such as racialism with negative word, this may generate the opposite meaning.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**August 09, 2018 at 5:31 pm**

Hi Lilya, I am not considering sentiment of a single word, but the entire tweet. For example, word2vec features for a single tweet have been generated by taking average of the word2vec vectors of the individual words in that tweet.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**August 09, 2018 at 5:37 pm**

Hi, Glad you liked it. I guess you are referring to the wordclouds generated for positive and negative sentiments. Please note that I have used train dataset for ploting these wordclouds wherein the data is labeled.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Mayank says:**August 09, 2018 at 5:54 pm**

Importing module nltk.tokenize.moses is raising ModuleNotFound error. Also, it doesn't seems to be there in NLTK3.3. Can anybody confirm?[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**August 09, 2018 at 6:54 pm**

Thanks Mayank for pointing it out. I have updated the code.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Jash says:**August 27, 2018 at 11:17 am**

Still, I cannot find the data file. Please help.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Aishwarya Singh says:**August 27, 2018 at 12:12 pm**

Hi Jash, Prateek has provided the link to the practice problem on datahack. Please register in the competition using the link provided. Once you do that, you will be able to download the dataset (train, test and submission files will be available after the problem statement at the bottom of the page). If you still face any issue, please let us know.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Gupta says:**August 28, 2018 at 3:08 pm**

Hi Prateek, I am getting NameError: name 'train' is not defined in this line- xtrain\_bow, xvalid\_bow, ytrain, yvalid = train\_test\_split(train\_bow, train['label'], random\_state=42, test\_size=0.3) I think you missed to mention how you separated and store the target variable.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**August 29, 2018 at 4:39 pm**

Hi, I am not getting this error. Make sure you have not missed any code. Thanks[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Ravinder Ahuja says:**September 20, 2018 at 1:15 pm**

Dear ITS NICE ARTICLE WITH GOOD EXPLANATION BUT I AM GETTING ERROR: ValueError: empty vocabulary; perhaps the documents only contain stop words. PLEASE HELP ME TO RESOLVE THIS. Thanks[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**September 20, 2018 at 2:29 pm**

Hi Ravinder, Which part of the code is giving you this error? Regards, Prateek[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Ravinder says:**September 21, 2018 at 1:03 am**

resolved....Thanks[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Ravinder Ahuja says:**September 21, 2018 at 1:20 am**

Hi not able to print word cloud showing error ValueError: We need at least 1 word to plot a word cloud, got 0.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Dilip Kumar says:**September 22, 2018 at 10:56 pm**

very nice explaination sir,this is really helpful sir[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Muhammad Younus says:**September 23, 2018 at 11:14 am**

Best article, you explain everything very nicely,Thanks[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Gaurav Rai says:**October 05, 2018 at 7:53 pm**

Hi Prateek, I am getting error for the sttiching together of tokens section: for i in range(len(tokenized\_tweet)): tokenized\_tweet[i] = ' '.join(tokenized\_tweet[i]) combi['tidy\_tweet'] = tokenized\_tweet I indented the code in the loop but still i am getting below error: for i in range(len(tokenized\_tweet)): tokenized\_tweet[i] = ' '.join(tokenized\_tweet[i]) combi['tidy\_tweet'] = tokenized\_tweet[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Gaurav Rai says:**October 05, 2018 at 9:57 pm**

Hi, For my previous comment i tried this and it worked: for i in range(len(tokenized\_tweet)): s = "" for j in tokenized\_tweet.iloc[i]: s += ''.join(j)+' ' tokenized\_tweet.iloc[i] = s.rstrip() Thanks for your time on this[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



tejeshwari says:**October 06, 2018 at 11:46 am**

Hi, I am registered on https://datahack.analyticsvidhya.com/contest/practice-problem-twitter-sentiment-analysis/#data\_dictionary, but still unable to download the twitter dataset.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**October 06, 2018 at 5:15 pm**

Great![**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**October 06, 2018 at 5:23 pm**

Hi Tejeshwari, you can find the download links just above the solution checker at the contest page.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Suriya says:**October 06, 2018 at 6:03 pm**

can u send me the full code please.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Aishwarya Singh says:**October 08, 2018 at 10:45 am**

The code is present in the article itself[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



humera hassan says:**October 21, 2018 at 10:43 pm**

for i in range(len(tokenized\_tweet)): s = “” for j in tokenized\_tweet.iloc[i]: s += ”.join(j)+’ ‘ tokenized\_tweet.iloc[i] = s.rstrip() i am getting error for this code as : File "", line 2 s = “” ^ IndentationError: expected an indented block[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**October 22, 2018 at 2:47 pm**

Hi, you have to indent after `for j in tokenized\_tweet.iloc[i]:`[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Ziza says:**November 03, 2018 at 5:04 pm**

Hi, In the beginning when you perform this step # remove twitter handles (@user) combi['tidy\_tweet'] = np.vectorize(remove\_pattern)(combi['tweet'], "@[\w]\*") Do you need to convert combi['tweet'] pandas.Series to string or byte-like object? I couldn't pass in a pandas.Series without converting it first![**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**November 03, 2018 at 5:15 pm**

Hi Ziza, The code is working fine at my end. I didn't convert combi[‘tweet’] to any other type.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Ziza says:**November 04, 2018 at 11:49 am**

Yeah, when I used your dataset everything worked just fine. I was actually trying that on another dataset, I guess I should pre-process those data. Thanks for your reply![**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Jingmiao Shen says:**November 08, 2018 at 12:21 am**

Great Great article!!![**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



pablo says:**November 19, 2018 at 6:34 pm**

Hi, excellent job with this article. I have started to learn machine learning to implement it in my django projects and this helped so much. I just have one thing to add. The stemmer that you used is behaving weird, i.e. changing 'this' to 'thi'. I have checked in the official repository and it is a known issue. So my advice would be to change it to stemming. It can be installed from pip, and you just use it like: `from stemming.porter2 import stem` stem('this') 'this' After changing to that stemmer the wordcloud started to look more accurate. Thanks again for the article![**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**November 20, 2018 at 3:18 pm**

Thanks Pablo for the feedback.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Zarief Marzuki Lao says:**December 08, 2018 at 8:30 am**

I was facing the same problem and was in a 'newbie-stuck' stage, where has all the s, i, e, y gone !!? Now I can proceed and continue to learn. This step by step tutorial is awesome. Many thanks to both Prateek & Pablo[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Vishwa Dadhania says:**December 31, 2018 at 1:39 pm**

Hi, Even after logging in I am not finding any link to download the dataset anywhere on the page. Is it because the practice problem competition is already over? Thanks & Regards[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Sharik says:**January 02, 2019 at 4:30 pm**

Hey, Prateek Even I am getting the same error. NameError: name 'train' is not defined And, even if you have a look at the code provided in the step 5 A) Building model using Bag-of-Words features. There is no variable declared as "train" it is either "train\_bow" or "test\_bow". So while splitting the data there is an error when the interpreter encounters "train['label']". Please look into it. Thanks :)[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**January 15, 2019 at 11:35 am**

Thanks Jingmiao[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**January 15, 2019 at 11:40 am**

Hi Sharik, I have read the train data in the beginning of the article. Please run the entire code.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Eesha Chinchwadkar says:**January 22, 2019 at 5:30 pm**

Hi Prateek, train\_bow = bow[:31962, :] test\_bow = bow[31962:, :] What is 31962 here? I am actually trying this on a different dataset to classify tweets into 4 affect categories. The length of my training set is 3960 and that of testing set is 3142.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**January 22, 2019 at 5:57 pm**

Hi Eesha, Here 31962 is the size of the training set. You may use 3960 instead. Regards[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Nidhi Sandilya says:**February 04, 2019 at 7:41 pm**

Hi Prateek, This is wonderfully written and carefully explained article, it is a very good read. Thank you for penning this down.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**February 06, 2019 at 11:24 am**

Thanks Nidhi :-)[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Anant Vignesh says:**February 11, 2019 at 2:21 am**

Hi Prateek, I just wanted to know where are you getting the label values? Where are you calculating it? Because if you are scrapping the tweets from twitter it does not come with that field. So how are you determining whether it is a positive or a negative tweet?[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**February 11, 2019 at 10:53 am**

Hi Anant, This dataset was manually labeled.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Caroline says:**February 12, 2019 at 4:01 pm**

Sir ..This was a good article i've gone through....Could you please share me the entire code so that i could use it as reference for my project.....[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Rathna Priya says:**February 12, 2019 at 4:05 pm**

Can you share your full working code with all the datasets needed[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**February 12, 2019 at 4:34 pm**

I have already shared the link to the full code at the end of the article. Please check.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**February 12, 2019 at 4:37 pm**

Hi Caroline, I have already shared the link to the full code at the end of the article. Please check.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**February 19, 2019 at 1:44 pm**

Hi Tom, The entire code has been shared in the end. Feel free to use it. Regards, Prateek Joshi[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Sk says:**March 03, 2019 at 9:03 am**

Beautiful article with great explanation! Thank you for your effort.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Unmesh kadam says:**March 03, 2019 at 3:13 pm**

Sir this is wonderful article, excellent work. Can we increase the F1 score?..plz suggest some method[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**March 03, 2019 at 5:22 pm**

I am glad you liked it.[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Gnanaprakash says:**March 03, 2019 at 5:40 pm**

WOW!!! Such a great article.. can you tell me how to categorize health related tweets like fever,malaria,dengue etc. instead of hate speech[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)



Prateek Joshi says:**March 03, 2019 at 7:12 pm**

Hi, You have to arrange health-related tweets first on which you can train a text classification model. That model would then be useful for your use case. Thanks and regards, Prateek[**Reply**](https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/#respond)

Leave a Reply**Your email address will not be published. Required fields are marked \***

Top of Form







Submit

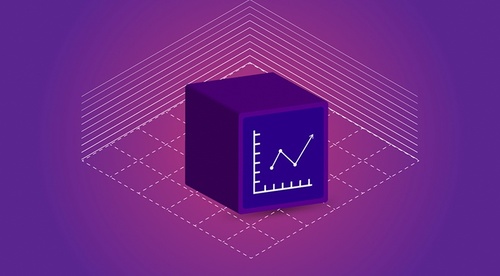
Bottom of Form

Top Resources

[[](https://www.analyticsvidhya.com/blog/2021/08/exploring-matplotlib-stylesheets-for-data-visualization/)](https://www.analyticsvidhya.com/blog/2021/08/exploring-matplotlib-stylesheets-for-data-visualization/)

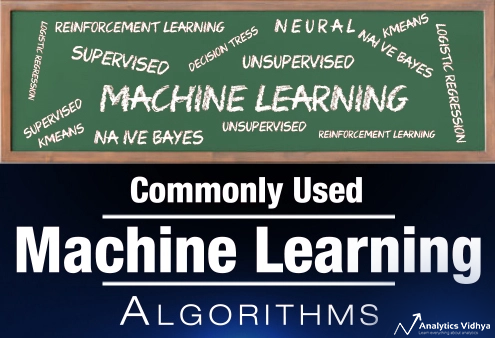
[Exploring Matplotlib Stylesheets For Data Visualization](https://www.analyticsvidhya.com/blog/2021/08/exploring-matplotlib-stylesheets-for-data-visualization/)

[rahul105 -](https://www.analyticsvidhya.com/blog/author/rahul105/)AUG 09, 2021

[[](https://www.analyticsvidhya.com/blog/2021/08/performance-comparision-of-regularized-and-unregularized-regression-models/)](https://www.analyticsvidhya.com/blog/2021/08/performance-comparision-of-regularized-and-unregularized-regression-models/)

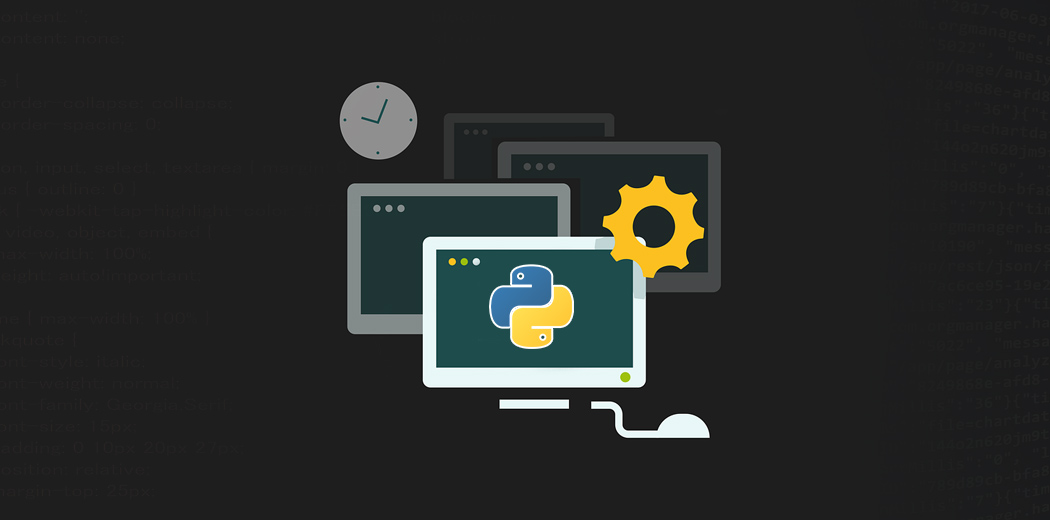
[Performance Comparision of Regularized and Unregularized Regression Models](https://www.analyticsvidhya.com/blog/2021/08/performance-comparision-of-regularized-and-unregularized-regression-models/)

[rahul105 -](https://www.analyticsvidhya.com/blog/author/rahul105/)AUG 03, 2021

[[](https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/)](https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/)

[Commonly used Machine Learning Algorithms (with Python and R Codes)](https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/)

[Sunil Ray -](https://www.analyticsvidhya.com/blog/author/sunil-ray/)SEP 09, 2017

[[](https://www.analyticsvidhya.com/blog/2020/09/object-oriented-programming/)](https://www.analyticsvidhya.com/blog/2020/09/object-oriented-programming/)

[Basic Concepts of Object-Oriented Programming in Python](https://www.analyticsvidhya.com/blog/2020/09/object-oriented-programming/)

[Himanshi Singh -](https://www.analyticsvidhya.com/blog/author/hs13/)SEP 01, 2020

[](https://www.analyticsvidhya.com/)

**Download App**

* **Analytics Vidhya**

[About Us](https://www.analyticsvidhya.com/about/)[Our Team](https://www.analyticsvidhya.com/team/)[Careers](https://www.analyticsvidhya.com/careers/)[Contact us](https://www.analyticsvidhya.com/contact/)

* **Data Scientists**

[Blog](https://www.analyticsvidhya.com/blog/)[Hackathon](https://datahack.analyticsvidhya.com/)[Discussions](https://discuss.analyticsvidhya.com/)[Apply Jobs](https://www.analyticsvidhya.com/jobs/)

* **Companies**

[Post Jobs](https://www.analyticsvidhya.com/corporate/)[Trainings](https://courses.analyticsvidhya.com/)[Hiring Hackathons](https://datahack.analyticsvidhya.com/)[Advertising](https://www.analyticsvidhya.com/contact/)

* **Visit us**

© Copyright 2013-2021 Analytics Vidhya.

[Privacy Policy](https://www.analyticsvidhya.com/privacy-policy/) [Terms of Use](https://www.analyticsvidhya.com/terms/) [Refund Policy](https://www.analyticsvidhya.com/refund-policy/)

Privacy Overview

This website uses cookies to improve your experience while you navigate through the website. Out of these, the cookies that are categorized as necessary are stored on your browser as they are essential for the working of basic functionalities of the ...

NecessaryAlways Enabled

Non-Necessary

SAVE & ACCEPT

:)