KURUNJI VENKATRAMANA GOWDA POLYTECHNIC SULLIA-574327

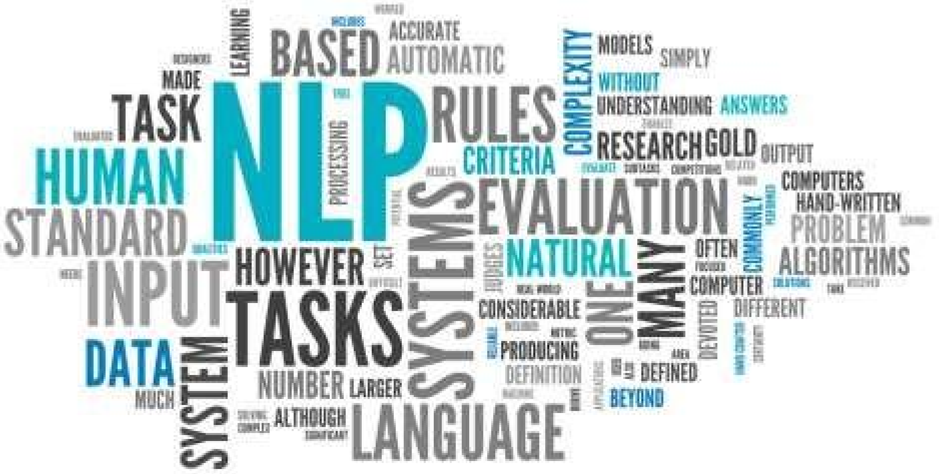
5TH SEMESTER

AI/ML WEEK-12

Natural Language Processing (NLP):

Introduction

Natural Language Processing (NLP) is a subpart of Artificial Intelligence that uses algorithms to understand and process human language. Various computational methods are used to process and analyze human language and a wide variety of real-life problems are solved using Natural Language Processing.



Using Natural Language Processing, we use machines by making them understand how human language works. Basically, we use text data and make computers analyze and process large quantities of such data. There is high demand for such data in today’s world as such data contains a vast amount of information and insight into business operations and profitability.

Importance of NLP :

Natural Language Processing has made things very easy in our modern life. Remember the time, when you typed the first few phrases of a question in Google, and Google guessed the remaining question. That is by NLP.

How Gmail now provides sample replies to the emails you receive, is also thanks to NLP. NLP has made our lives a lot easier in the technology we use in our day-to-day lives.

There are a lot of uses of NLP. Let us have a look at some interesting NLP uses.

NLP Uses :

Grammar Correction Tools

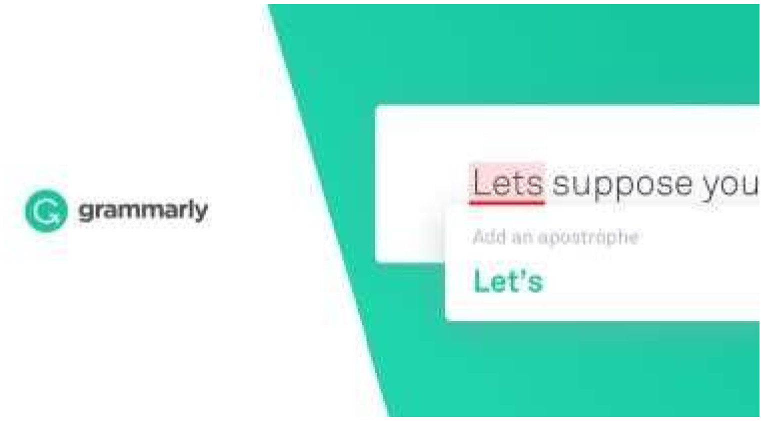
Grammar correction tools are one of the most widely used applications of NLP. Such tools check errors in our text and give suggestions on which corrections are to be made. These tools are already fed with data about correct grammar and know between correct and incorrect usage.

Grammarly is one of the most common tools in this scenario. Grammar correction tools are of immense use and utility to all. Everyone starting from students to senior executives can use them to make their writing better and crisp.

With grammar tools, the quality of writing improves drastically and many people are interested in the paid version of the product, leading to better revenues.

Grammarly is a cloud-based grammar correction tool. According to them, their team of linguists and deep learning engineers design algorithms that learn the rules and patterns of good writing, by analyzing millions of sentences from research text.

It also learns with data, every time a user accepts or ignores a suggestion given by Grammarly, the AI gets smarter. The exact functioning of the AI is not revealed, but surely it uses a lot of NLP techniques.

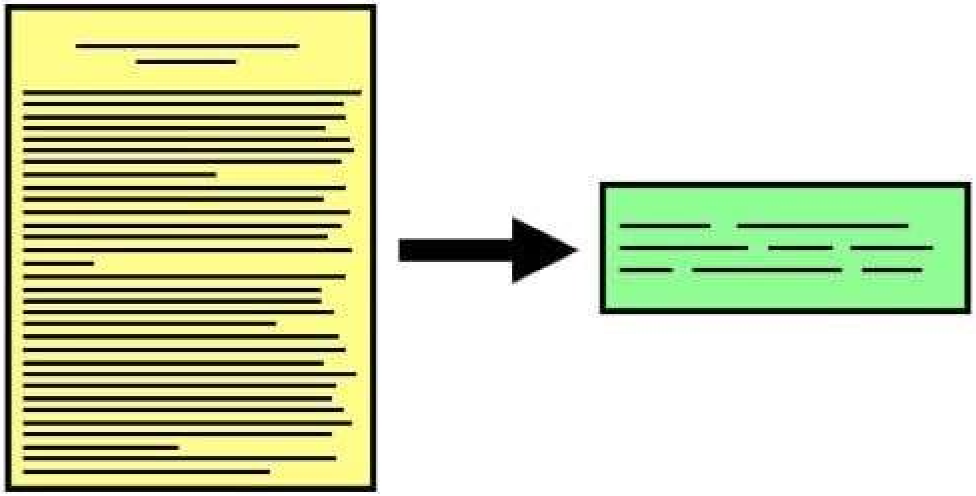


Text Summarisation

Text Summarisation is an interesting application of NLP. Often, you want to submit an essay or write an article, but there seems to be a limit to the length of the text you can submit.

Text Summarization is the process of condensing a long piece of text into a shorter version, preserving the basic idea of the text and still containing all the key points.

Manual text summarization is often very expensive and time-consuming and a tedious job. The large amount of text data available in today's modern world of big data is enormous. This huge amount of text data has a large potential for business growth and useful analytics. Automatic text summarization has huge potential these days.



(Image Source: https://www.kdnuggets.com/2019/11/getting-started-automated-text-summarization.html)

Text Compactor is a good and useful tool for text summarization. Do check out. Text Summaries reduce reading time, make the selection process easier in many cases. With better computational resources and better research, the summary can be as good as written by a human. There are mainly two types of Text Summarization :

1. Extractive Methods.
2. Abstractive Methods.

In Extractive methods, algorithms use sentences and phrases from the source text to create the summary. The algorithm uses word frequency, the relevance of phrases, and other parameters to arrive at the summary.

Summarization by abstractive methods is a way of summary creation by the generation of new sentences and phrases as compared to the source document. This type of method is often more difficult to execute and needs more advanced approaches like Deep Learning.

Implementing The Text Summarizer

One of the most simple ways to implement a text summarizer is to create a summarizer based on using the sentences with the most weightage. Weightage is calculated by taking words with the most usage in the original text. The code :

```
import docx2txt
import re
import heapq
import nltk
import docx
text= docx2txt.process("Text Input.docx")
#Removing thw Square Brackets and Extra Spaces
```

```
article_text = re.sub(r'[[0-9]*]', ' ', text)

article_text = re.sub(r's+', ' ',article_text )

#Sentence tokenization

sentence_list = nltk.sent_tokenize(article_text)

#Now we need to find the frequency of each word before it was tokenized into sentences

#Word frequency

#Also we are removing the stopwords from the text we are using

stopwords = nltk.corpus.stopwords.words('english')

#Storing the word frequencies in a dictionary

word_frequencies = {}

for word in nltk.word_tokenize(article_text):

    if word not in stopwords:

        if word not in word_frequencies.keys():

            word_frequencies[word] = 1

        else:

            word_frequencies[word] += 1

#Now we need to find the weighted frequency

#We shall divide the number of occurances of all the words by the frequency of the most
occurring word


maximum_frequncy = max(word_frequencies.values())

for word in word_frequencies.keys():

    word_frequencies[word] = (word_frequencies[word]/maximum_frequncy)

#Using relative word frequency, not absolute word fequency so as to distribute the values from
0-1

#Calculating Sentence Scores

#Scores for each sentence obtained by adding weighted frequencies of the words that occur in
that particular sentence.

sentence_scores = {}

for sent in sentence_list:

    for word in nltk.word_tokenize(sent.lower()):
        if word in word_frequencies.keys():
            if len(sent.split(' ')) < 25:
                if sent not in sentence_scores.keys():
                    sentence_scores[sent] = word_frequencies[word]
                else:
                    sentence_scores[sent] += word_frequencies[word]
#To summarize the article we are going to take the sentences with 10 highest scores
#This parameter can be changed according to the length of the text
summary_sent = heapq.nlargest(10, sentence_scores, key=sentence_scores.get)
summary = ' '.join(summary_sent)
print(summary)
mydoc = docx.Document()
mydoc.add_paragraph(summary)
mydoc.save("Text Output.docx")
```

The overall logic is simple. The code extracts the input code from a text document. Text is processes and stopwords are removed. Then the word frequencies are calculated in each sentence and sentences are given a sentence score. The number of sentences in the summary can also be decided. Ultimately the summary text is written to another doc file.

The sample below taken from the Ferrari Wikipedia page shows an implementation of the code.

A sample input fed into the code:

```
Enzo Ferrari was not initially interested in the idea of producing road cars when he formed
Scuderia Ferrari in 1929, with headquarters in Modena. Scuderia Ferrari (pronounced
[skudeˈriːa]) literally means "Ferrari Stable" and is usually used to mean "Team Ferrari." Ferrari
bought,[citation needed] prepared, and fielded Alfa Romeo racing cars for gentleman drivers,
functioning as the racing division of Alfa Romeo. In 1933, Alfa Romeo withdrew its in-house
racing team and Scuderia Ferrari took over as its works team:[1] the Scuderia received Alfa's
Grand Prix cars of the latest specifications and fielded many famous drivers such as Tazio
Nuvolari and Achille Varzi. In 1938, Alfa Romeo brought its racing operation again in-house,
forming Alfa Corse in Milan and hired Enzo Ferrari as manager of the new racing department;
therefore the Scuderia Ferrari was disbanded.[1]
```

The first vehicle made with the Ferrari name was the 125 S. Only two of this small two-seat sports/racing V12 car were made. In 1949, the 166 Inter was introduced marking the company's significant move into the grand touring road car market. The first 166 Inter was a four-seat (2+2) berlinetta coupe with body work designed by Carrozzeria Touring Superleggera. Road cars quickly became the bulk of Ferrari sales.

The early Ferrari cars typically featured bodywork designed and customised by independent coachbuilders such as Pininfarina, Scaglietti, Zagato, Vignale and Bertone.

The original road cars were typically two-seat front-engined V12s. This platform served Ferrari very well through the 1950s and 1960s. In 1968 the Dino was introduced as the first two-seat rear mid-engined Ferrari. The Dino was produced primarily with a V6 engine, however, a V8 model was also developed. This rear mid-engine layout would go on to be used in many Ferraris of the 1980s, 1990s and to the present day. Current road cars typically use V8 or V12 engines, with V8 models making up well over half of the marque's total production. Historically, Ferrari has also produced flat 12 engines.

In September 1939, Ferrari left Alfa Romeo under the provision he would not use the Ferrari name in association with races or racing cars for at least four years.[1] A few days later he founded Auto Avio Costruzioni, headquartered in the facilities of the old Scuderia Ferrari.[1] The new company ostensibly produced machine tools and aircraft accessories. In 1940, Ferrari produced a race car – the Tipo 815, based on a Fiat platform. It was the first Ferrari car and debuted at the 1940 Mille Miglia, but due to World War II it saw little competition. In 1943, the Ferrari factory moved to Maranello, where it has remained ever since. The factory was bombed by the Allies and subsequently rebuilt including works for road car production.

125 S replica

166 MM Touring Barchetta

The first series produced Ferrari, the 1958 250 GT Coupé

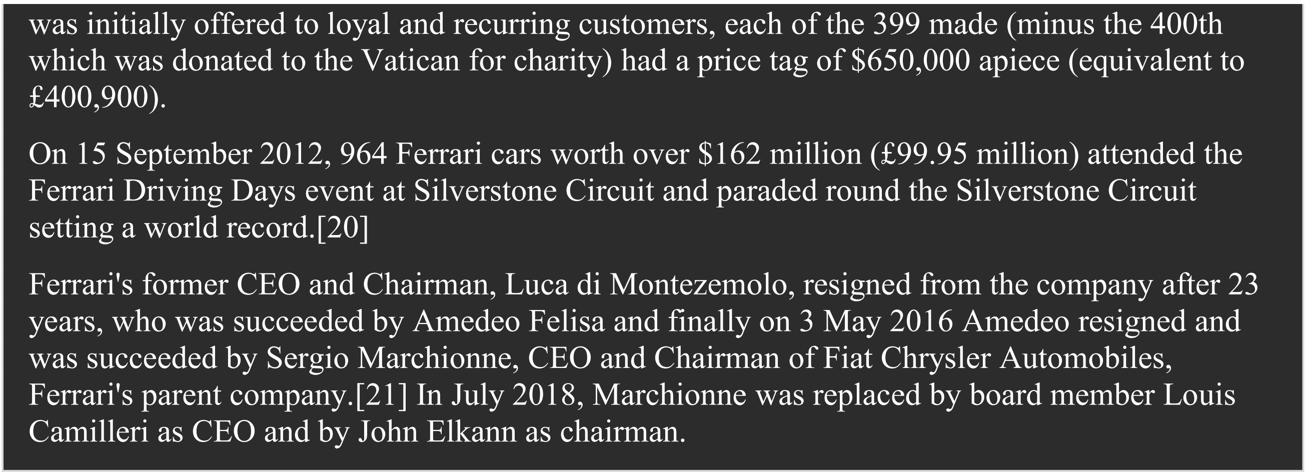
The first Ferrari-badged car was the 1947 125 S, powered by a 1.5 L V12 engine;[1] Enzo Ferrari reluctantly built and sold his automobiles to fund Scuderia Ferrari.[18]

The Scuderia Ferrari name was resurrected to denote the factory racing cars and distinguish them from those fielded by customer teams.

In 1960 the company was restructured as a public corporation under the name SEFAC S.p.A. (Società Esercizio Fabbriche Automobili e Corse).[19]

Early in 1969, Fiat took a 50% stake in Ferrari. An immediate result was an increase in available investment funds, and work started at once on a factory extension intended to transfer production from Fiat's Turin plant of the Ferrari engined Fiat Dino. New model investment further up in the Ferrari range also received a boost.

In 1988, Enzo Ferrari oversaw the launch of the Ferrari F40, the last new Ferrari launched before his death later that year. In 1989, the company was renamed Ferrari S.p.A.[19] From 2002 to 2004, Ferrari produced the Enzo, their fastest model at the time, which was introduced and named in honor of the company's founder, Enzo Ferrari. It was to be called the F60, continuing on from the F40 and F50, but Ferrari was so pleased with it, they called it the Enzo instead. It

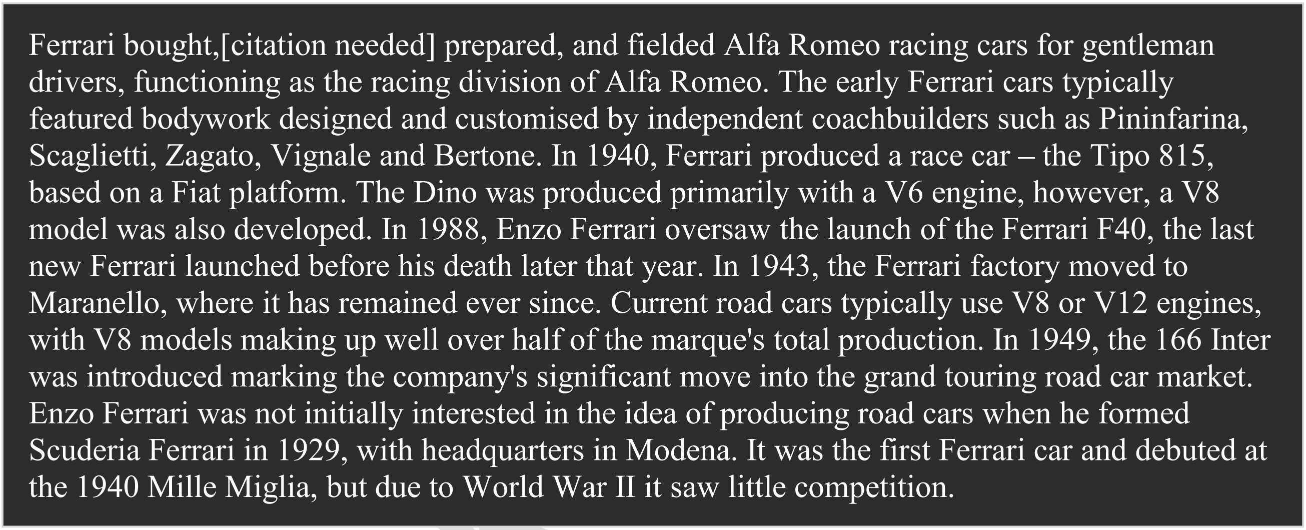


was initially offered to loyal and recurring customers, each of the 399 made (minus the 400th which was donated to the Vatican for charity) had a price tag of $650,000 apiece (equivalent to £400,900).

On 15 September 2012, 964 Ferrari cars worth over $162 million (£99.95 million) attended the Ferrari Driving Days event at Silverstone Circuit and paraded round the Silverstone Circuit setting a world record.[20]

Ferrari's former CEO and Chairman, Luca di Montezemolo, resigned from the company after 23 years, who was succeeded by Amedeo Felisa and finally on 3 May 2016 Amedeo resigned and was succeeded by Sergio Marchionne, CEO and Chairman of Fiat Chrysler Automobiles, Ferrari's parent company.[21] In July 2018, Marchionne was replaced by board member Louis Camilleri as CEO and by John Elkann as chairman.

The output:



```
Ferrari bought,[citation needed] prepared, and fielded Alfa Romeo racing cars for gentleman
drivers, functioning as the racing division of Alfa Romeo. The early Ferrari cars typically
featured bodywork designed and customised by independent coachbuilders such as Pininfarina,
Scaglietti, Zagato, Vignale and Bertone. In 1940, Ferrari produced a race car – the Tipo 815,
based on a Fiat platform. The Dino was produced primarily with a V6 engine, however, a V8
model was also developed. In 1988, Enzo Ferrari oversaw the launch of the Ferrari F40, the last
new Ferrari launched before his death later that year. In 1943, the Ferrari factory moved to
Maranello, where it has remained ever since. Current road cars typically use V8 or V12 engines,
with V8 models making up well over half of the marque's total production. In 1949, the 166 Inter
was introduced marking the company's significant move into the grand touring road car market.
Enzo Ferrari was not initially interested in the idea of producing road cars when he formed
Scuderia Ferrari in 1929, with headquarters in Modena. It was the first Ferrari car and debuted at
the 1940 Mille Miglia, but due to World War II it saw little competition.
```

The code determined the 10 most suitable sentences and used them to form the summary. This summary is made using an Extractive method, as the summary contains sentences from the original text.

Text Analytics

Text Analytics is the process of gathering useful data and insights from text data. Businesses often have a large amount of data at their disposal, examples of text data would include customer product reviews, chatbot data, customer suggestion mails, and more.

Text analytics can be used to understand and identify data patterns and make business decisions. These methods include word/phase-frequency calculation, word cloud generation, sentiment analysis, and others. Also, it gives a good way to work with a large amount of text data.

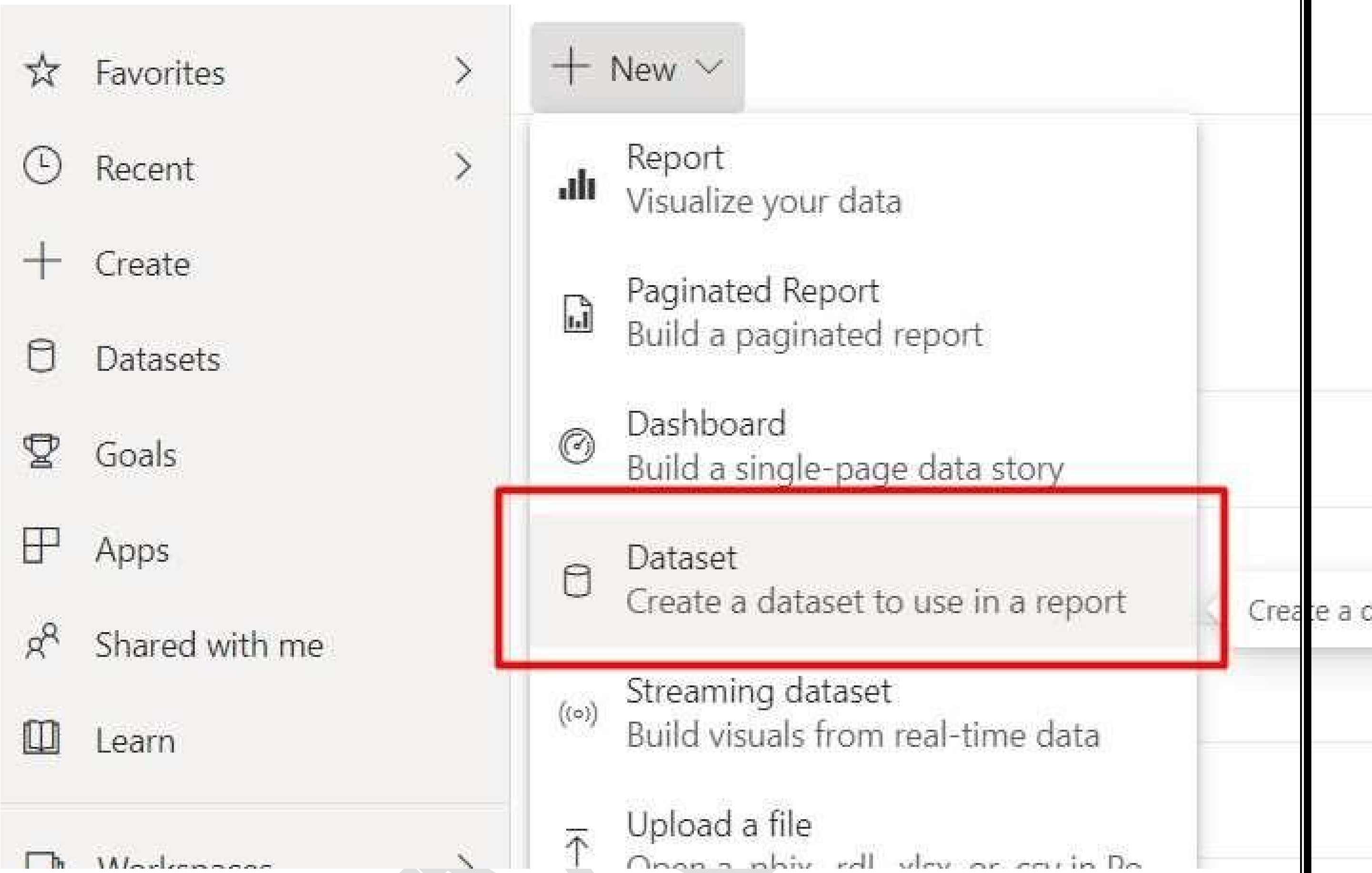
In the process of Text analytics, the source text or documents are processed, then various NLP methods are applied to them. Firstly, useless punctuation and symbols are removed. If the text happens to be web-scraped, there will be a lot of HTML, which has to be cleaned. Punctuation cleaning methods are then applied.

Then stopwords are to be removed. Stopwords are the most common words in a language, and often do not portray any sentiment. Stopwords in English are and, on, the, that, at, is, in, etc. There are various ways and algorithms to remove these stopwords from our text. Stemming and Lemmatization are also among the important steps in cleaning the text for analytics. With all the basic NLP techniques, the text becomes ready to be processed and worked upon.

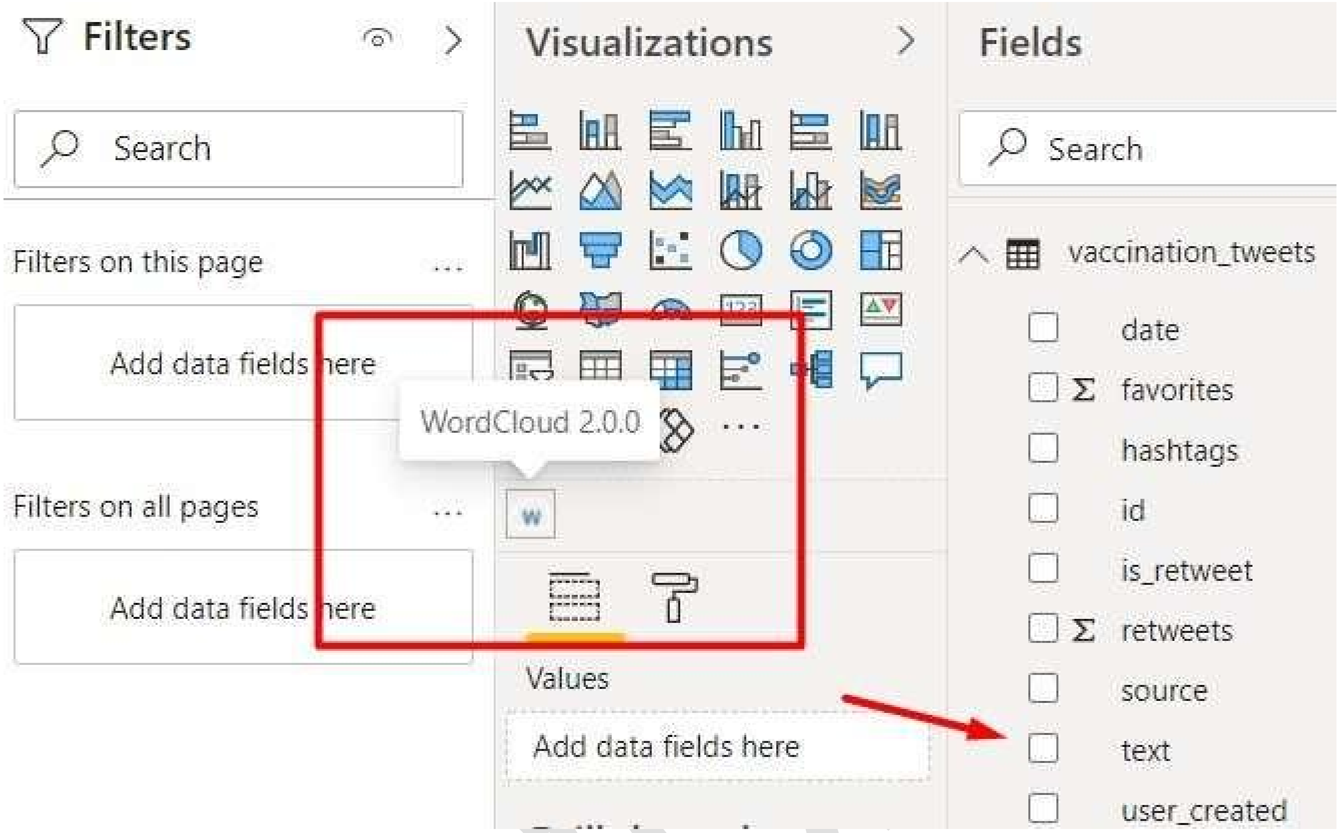
Text Analytics using Power BI

Let us work on some text analytics using Power BI. The dataset we are taking is the Pfizer and BioNTech Vaccine Tweets dataset from Kaggle. Dataset link: Pfizer Vaccine Tweets.

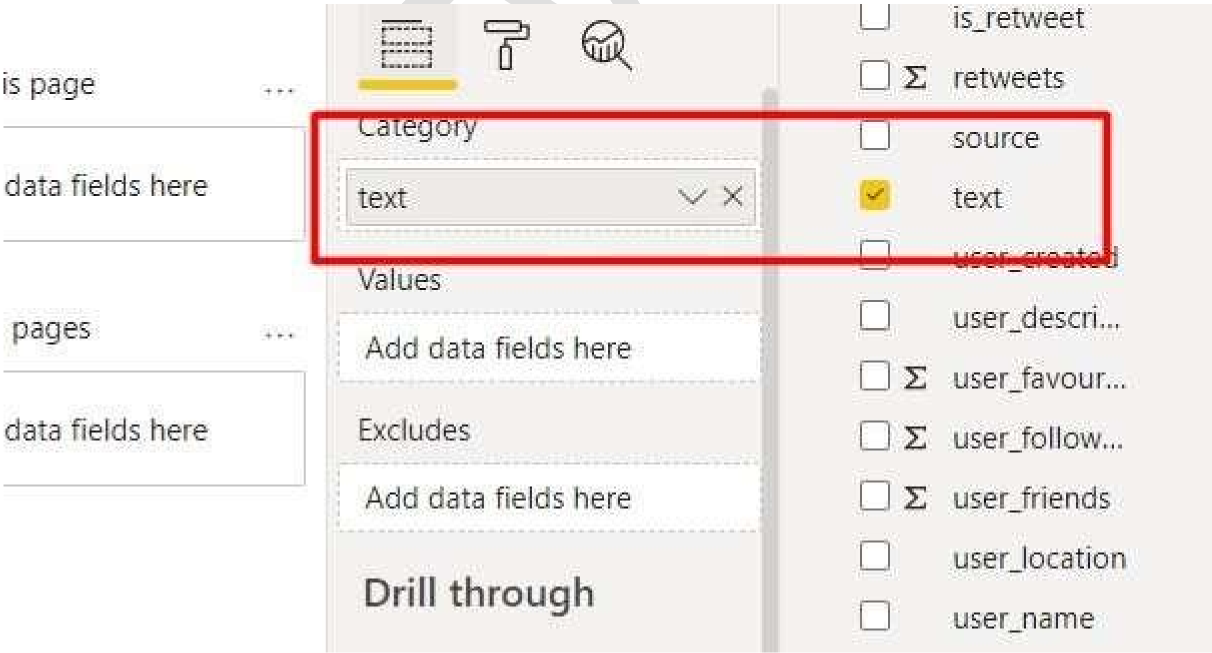
After downloading the data, we add the data to the Power BI service workspace.



Now we shall be working with one of the most simple and easy to implement Text Analytics method: A word cloud.



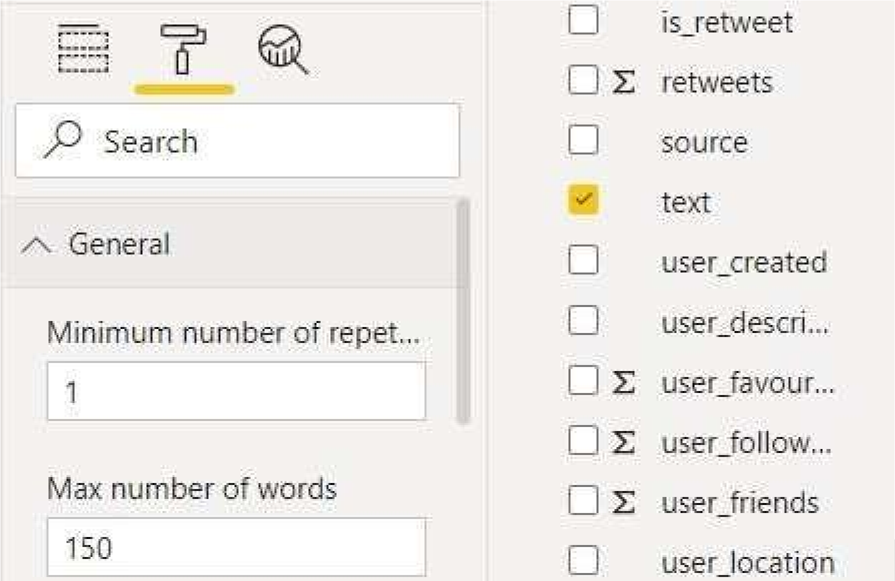
We drag and drop the WordCloud visual to the workspace.



Category
text

☑ text

Now, as the “text” field contains the tweet text, we will drag and drop it into the Category area. One of the best things about the Word Cloud visual is that the frequency of the words decides which words will be in the Word Cloud and the size of the words is decided by the frequency.



Search

General

Minimum number of repet...

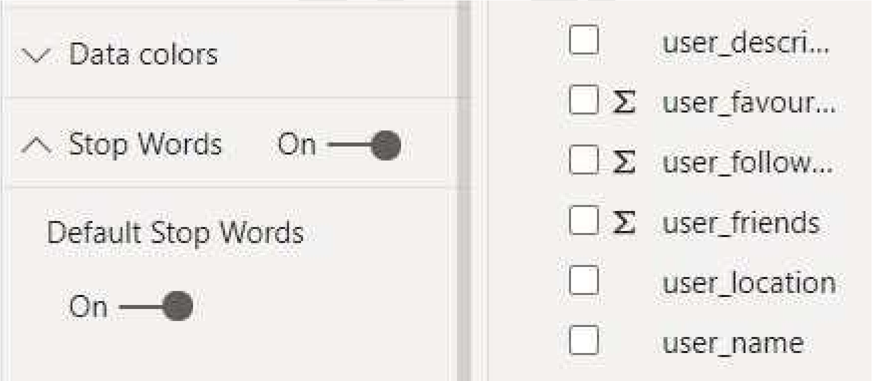
1

Max number of words

150

☐ is_retweet
☐ Σ retweets
☐ source
☑ text
☐ user_created
☐ user_descri...
☐ Σ user_favour...
☐ Σ user_follow...
☐ Σ user_friends
☐ user_location

We set the maximum number of words to 150. This indicates that the top 150 words which appeared the most will be present in the word cloud.



Data colors

Stop Words On

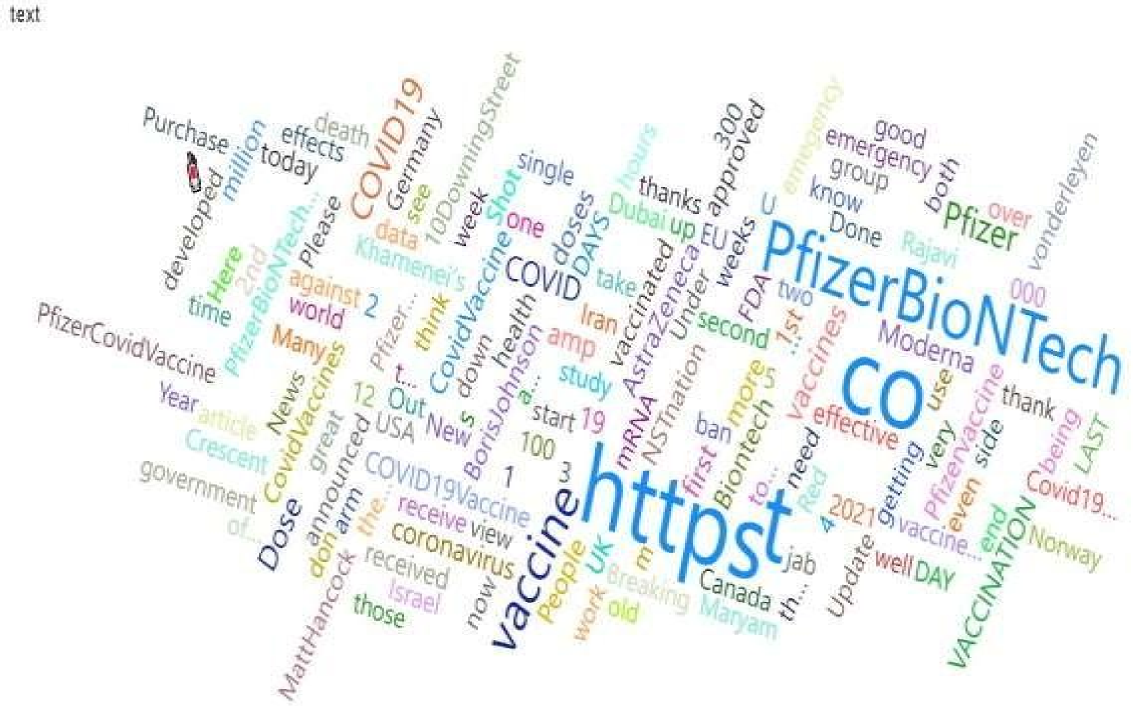
Default Stop Words

On

☐ user_descri...
☐ Σ user_favour...
☐ Σ user_follow...
☐ Σ user_friends
☐ user_location
☐ user_name

Now we remove the stopwords from the text. Stopwords are the most common words in a language that are needed for basic grammar and sentence structure. In English, common stopwords are “the”, “is”, “but” , “is”, “at” etc.

Now with the stopwords removed, the Word Cloud looks like this :



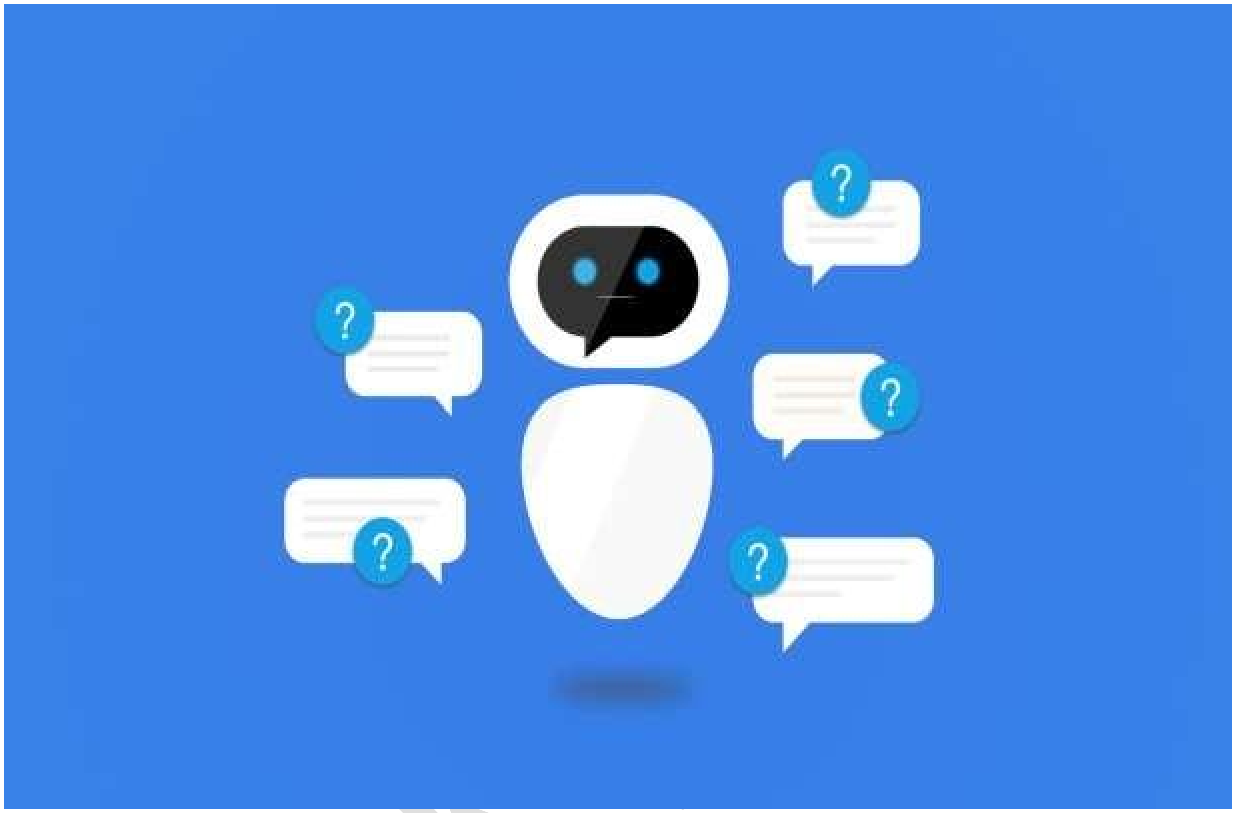
Here we can see a wide variety of vaccine technicalities and other stuff. The point to be noted is that “HTTPS” appears the largest as users might have shared websites and such, which led to it appearing the most.

Chatbots

Businesses need to have a strong customer helpline and support network. Chatbots are an integral part of a strong customer support network. Virtual assistants and chatbots are part of most online services and apps these days.

Chatbots have Natural Language Generation capabilities via which they can converse with a human customer or client and solve their problem or understand their problem before a human executive can take over. Chatbots are trained with probable questions and answers to those questions. Companies like Zomato have efficient chatbots that can solve a wide variety of queries.

Chatbots can make the user experience easier and more convenient, making the whole business more efficient. Automated chatbots can stay online 24/7, all 365 days of the year, something which human support executives can never do (a single person considered).



Chatbots are very efficient in capturing leads and converting them into customers for the business. With better technology, chatbots are getting more and more cost-effective and easy to interact with. The data fed to the chatbot is text data in form of user queries and user doubts.

The exact functioning of different chatbots might be different, but all follow a database or algorithm in which, when the user input is passed, a response is given which solves a business problem.

Market Research and Market Intelligence

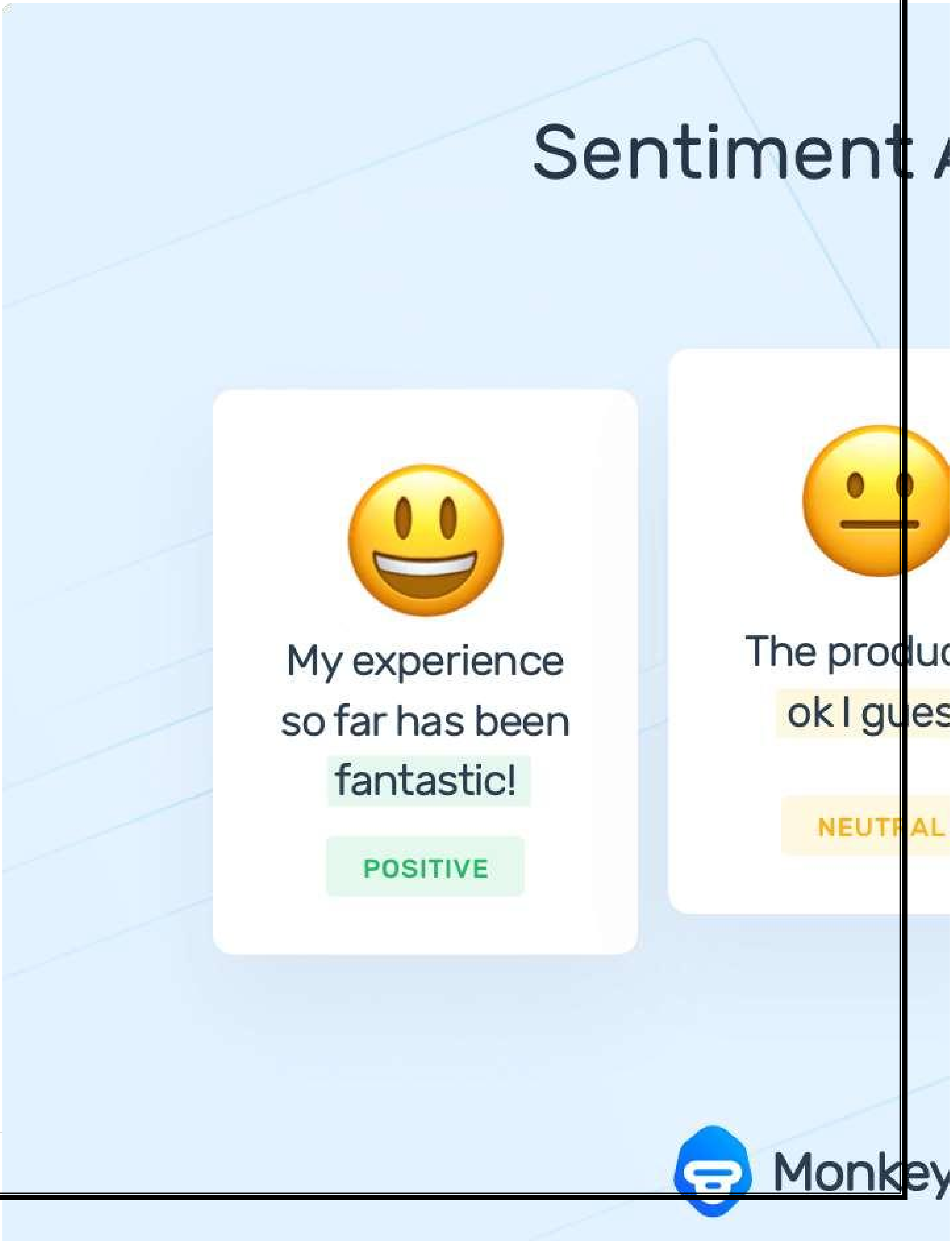
Companies use various NLP methods to analyze news and happenings in the market in an attempt to stay ahead of their competition. NLP tools monitor all the press releases, news reports, and competitor’s social media handles to get an idea of the market. With these data, companies and adjust their strategy to gain an edge over their competition.

Topic Modelling and Content Themes

Often companies want to gain more and more traffic from SEO to their websites. A good content strategy is important for that. Topic extraction can help in the identification of the most well-performing content on the internet for the marketing teams to decide on. Companies can understand audience intentions and use the data to better serve the needs of customers and audience. Using such data, better content themes can be identified and lead to better marketing strategies.

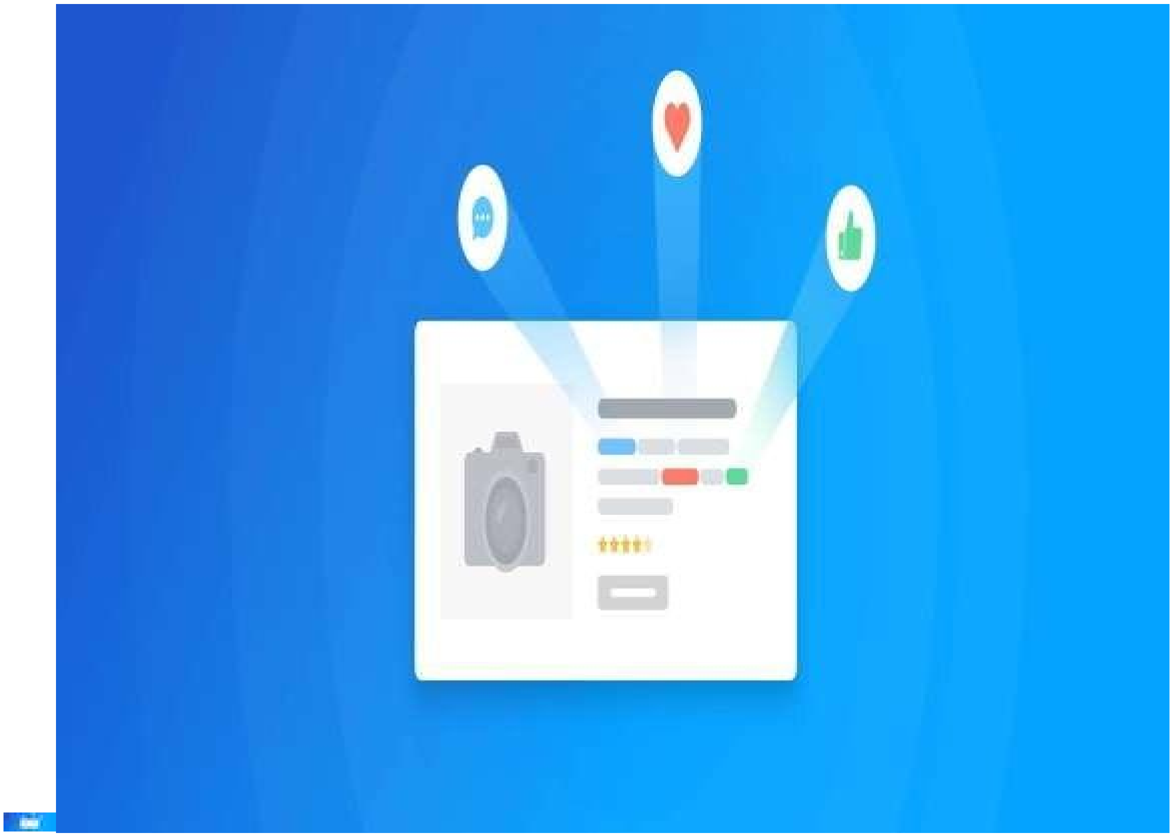
Email Classification

It is not a new thing today that our inbound emails get classified into our primary inbox, promotions, and spam inbox. Ever wondered? how does it happen? Well, NLP is at play here as well. Basically, it is a classification problem. Based on old data and other parameters, the AI is trained to identify the type of inbound mail. Spam mail tends to have a lot of irrelevant messages and unclear outbound links. Similarly, commercial and promotional emails tend to have promotional content, like coupons, discount offers. With the help of NLP, such messages are identified. The AI, based on the text content, can make the classification.


Sentiment A
My experience so far has been fantastic!
POSITIVE
The produc
ok I gues
NEUTRAL
Monkey

Sentiment analysis (or opinion mining) is a natural language processing (NLP) technique used to determine whether data is positive, negative or neutral. Sentiment analysis is often performed on textual data to help businesses monitor brand and product sentiment in customer feedback, and understand customer needs.

What Is Sentiment Analysis?



Sentiment analysis is the process of detecting positive or negative sentiment in text. It’s often used by businesses to detect sentiment in social data, gauge brand reputation, and understand customers.

Types of Sentiment Analysis

Sentiment analysis focuses on the polarity of a text (*positive, negative, neutral*) but it also goes beyond polarity to detect specific feelings and emotions (*angry, happy, sad*, etc), urgency (*urgent, not urgent*) and even intentions (*interested v. not interested*).

Depending on how you want to interpret customer feedback and queries, you can define and tailor your categories to meet your sentiment analysis needs. In the meantime, here are some of the most popular types of sentiment analysis:

Graded Sentiment Analysis

If polarity precision is important to your business, you might consider expanding your polarity categories to include different levels of positive and negative:

- Very positive
- Positive
- Neutral
- Negative
- Very negative

This is usually referred to as graded or fine-grained sentiment analysis, and could be used to interpret 5-star ratings in a review, for example:

- Very Positive = 5 stars
- Very Negative = 1 star

Emotion detection

Emotion detection sentiment analysis allows you to go beyond polarity to detect emotions, like happiness, frustration, anger, and sadness.

Many emotion detection systems use lexicons (i.e. lists of words and the emotions they convey) or complex machine learning algorithms.

One of the downsides of using lexicons is that people express emotions in different ways. Some words that typically express anger, like *bad* or *kill* (e.g. *your product is so bad* or *your customer support is killing me*) might also express happiness (e.g. *this is bad ass* or *you are killing it*).

Aspect-based Sentiment Analysis

Usually, when analyzing sentiments of texts you'll want to know which particular aspects or features people are mentioning in a positive, neutral, or negative way.

That's where aspect-based sentiment analysis can help, for example in this product review: *"The battery life of this camera is too short"*, an aspect-based classifier would be able to determine that the sentence expresses a negative opinion about the battery life of the product in question.

Multilingual sentiment analysis

Multilingual sentiment analysis can be difficult. It involves a lot of preprocessing and resources. Most of these resources are available online (e.g. sentiment lexicons), while others need to be created (e.g. translated corpora or noise detection algorithms), but you'll need to know how to code to use them.

Alternatively, you could detect language in texts automatically with a language classifier, then train a custom sentiment analysis model to classify texts in the language of your choice.

Why Is Sentiment Analysis Important?

Since humans express their thoughts and feelings more openly than ever before, sentiment analysis is fast becoming an essential tool to monitor and understand sentiment in all types of data.

Automatically analyzing customer feedback, such as opinions in survey responses and social media conversations, allows brands to learn what makes customers happy or frustrated, so that they can tailor products and services to meet their customers' needs.

For example, using sentiment analysis to automatically analyze 4,000+ open-ended responses in your customer satisfaction surveys could help you discover why customers are happy or unhappy at each stage of the customer journey.

Maybe you want to track brand sentiment so you can detect disgruntled customers immediately and respond as soon as possible. Maybe you want to compare sentiment from one quarter to the next to see if you need to take action. Then you could dig deeper into your qualitative data to see why sentiment is falling or rising.

The overall benefits of sentiment analysis include:

- **Sorting Data at Scale**

Can you imagine manually sorting through thousands of tweets, customer support conversations, or surveys? There's just too much business data to process manually. Sentiment analysis helps businesses process huge amounts of unstructured data in an efficient and cost-effective way.

- **Real-Time Analysis**

Sentiment analysis can identify critical issues in real-time, for example is a PR crisis on social media escalating? Is an angry customer about to churn? Sentiment analysis models can help you immediately identify these kinds of situations, so you can take action right away.

- **Consistent criteria**

It's estimated that people only agree around 60-65% of the time when determining the sentiment of a particular text. Tagging text by sentiment is highly subjective, influenced by personal experiences, thoughts, and beliefs.

By using a centralized sentiment analysis system, companies can apply the same criteria to all of their data, helping them improve accuracy and gain better insights.

The applications of sentiment analysis are endless. So, to help you understand how sentiment analysis could benefit your business, let's take a look at some examples of texts that you could analyze using sentiment analysis.

Then, we'll jump into a real-world example of how Chewy, a pet supplies company, was able to gain a much more nuanced (and useful!) understanding of their reviews through the application of sentiment analysis.

Sentiment Analysis Examples

To understand the goal and challenges of sentiment analysis, here are some examples:

Basic examples of sentiment analysis data

- Netflix has the best selection of films
- Hulu has a great UI
- I dislike like the new crime series
- I hate waiting for the next series to come out

More challenging examples of sentiment analysis

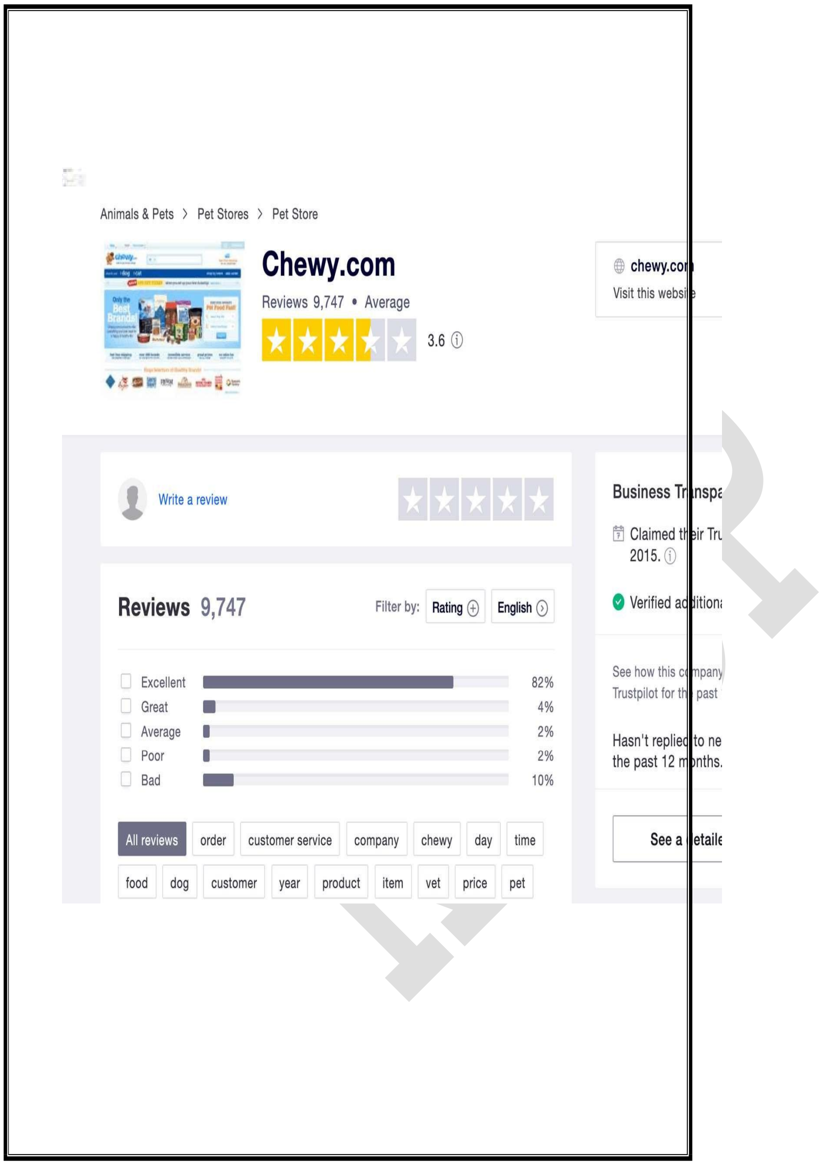
- I *do not dislike* horror movies. (phrase with negation)
- Disliking horror movies is *not* uncommon. (negation, inverted word order)
- *Sometimes* I really hate the show. (adverbial modifies the sentiment)
- I love having to wait two months for the next series to come out! (sarcasm)
- The final episode was surprising with a *terrible* twist at the end (negative term used in a positive way)
- The film was easy to watch but I would not recommend it to my friends. (difficult to categorize)
- I *LOL'd* at the end of the cake scene (often hard to understand new terms)

Now, let's take a look at some real reviews on Trustpilot and see how MonkeyLearn's sentiment analysis tools fare when it comes to recognizing and categorizing sentiment.

Case Study: Sentiment analysis on TrustPilot Reviews

Chewy is a pet supplies company – an industry with no shortage of competition, so providing a superior customer experience (CX) to their customers can be a massive difference maker.

For this reason, online reviews can be an extremely valuable source of information to gain customer insights to improve their CX. Chewy has thousands of reviews in TrustPilot, this is what their review archive looks like:



Animals & Pets > Pet Stores > Pet Store

Chewy.com

Reviews 9,747 • Average

3.6

chewy.co

Visit this websi e

Write a review

Business Tr nspa

Claimed th ir Tru 2015.

Verified ad itiona

See how this c mpany Trustpilot for th past

Hasn't replied to ne the past 12 m nths.

See a etaile

Reviews 9,747

Filter by: Rating English

□ Excellent	82%
□ Great	4%
□ Average	2%
□ Poor	2%
□ Bad	10%

All reviews | order | customer service | company | chewy | day | time | food | dog | customer | year | product | item | vet | price | pet

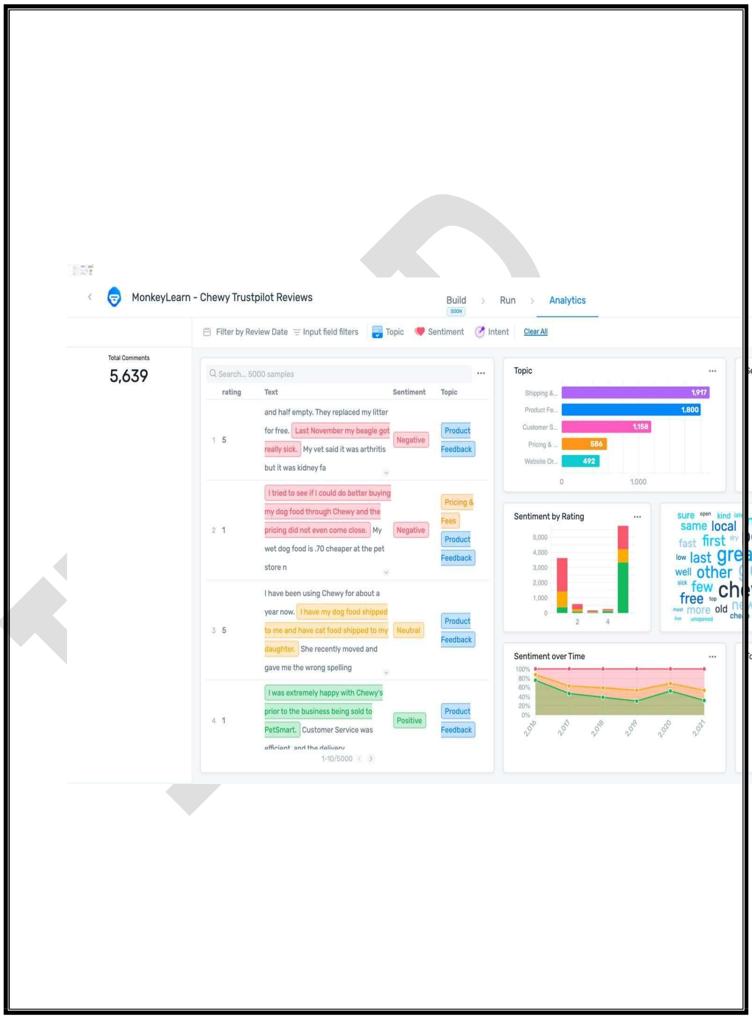
Via [*TrustPilot*]

It is easy to draw a *general* conclusion about Chewy's relative success from this alone - 82% of responses being excellent is a great starting place.

But TrustPilot's results alone fall short if Chewy's goal is to improve its services. This perfunctory overview fails to provide actionable insight, the cornerstone, and end goal, of effective sentiment analysis.

If Chewy wanted to unpack the *what* and *why* behind their reviews, in order to further improve their services, they would need to analyze each and every negative review at a granular level.

But with sentiment analysis tools, Chewy could plug in their 5,639 (at the time) TrustPilot reviews to gain instant sentiment analysis insights.


MonkeyLearn - Chewy Trustpilot Reviews

Total Comments
5,639

	rating	Text	Sentiment	Topic
1	5	and half empty. They replaced my litter for free. Last November my beagle got really sick. My vet said it was arthritis but it was kidney fa	Negative	Product Feedback
2	1	I tried to see if I could do better buying my dog food through Chewy and the pricing did not even come close. My wet dog food is .70 cheaper at the pet store n	Negative	Pricing & Fees Product Feedback
3	5	I have been using Chewy for about a year now. I have my dog food shipped to me and have cat food shipped to my daughter. She recently moved and gave me the wrong spelling	Neutral	Product Feedback
4	1	I was extremely happy with Chewy's prior to the business being sold to PetSmart. Customer Service was efficient and the delivery	Positive	Product Feedback

Chewy TrustPilot Reviews Sample

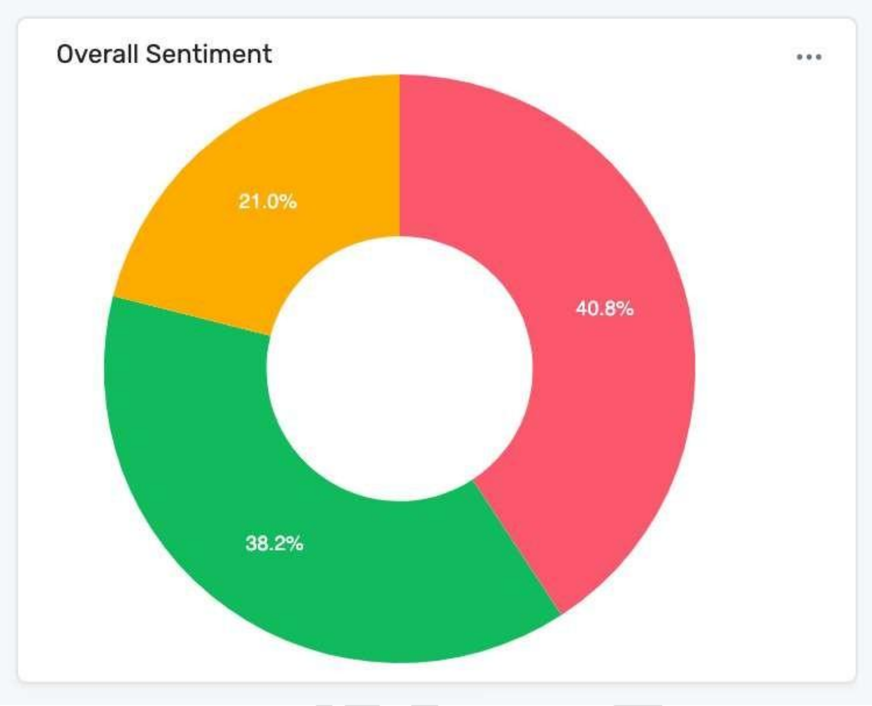
Feel free to click this link to peruse the results at your leisure - as this sample dashboard is a public demo, you can click through and explore the inputs and filters at work yourself.

While there is a ton more to explore, in this breakdown we are going to focus on four sentiment analysis data visualization results that the dashboard has visualized for us.

1. Overall Sentiment
2. Sentiment over Time
3. Sentiment by Rating
4. Sentiment by Topic

1. Overall sentiment

We’ll begin by pulling the relevant graphic from the above dashboard.



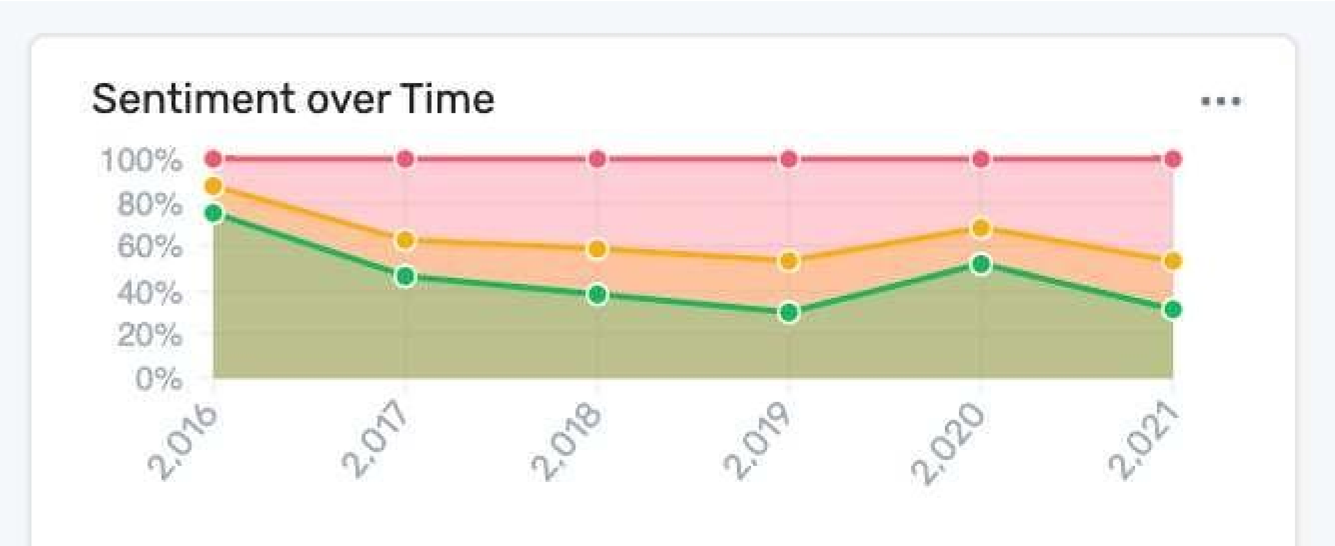
You’ll notice that these results are very different from TrustPilot’s overview (82% excellent, etc). This is because MonkeyLearn’s sentiment analysis AI performs advanced sentiment analysis, parsing through each review sentence by sentence, word by word.

What you are left with is an accurate assessment of everything customers have written, rather than a simple tabulation of stars. This analysis can point you towards friction points much more accurately and in much more detail.

Read up on the mechanics of how sentiment analysis works below.

2. Sentiment over time

Here’s our handy-dandy sentiment over time graph, blown up:

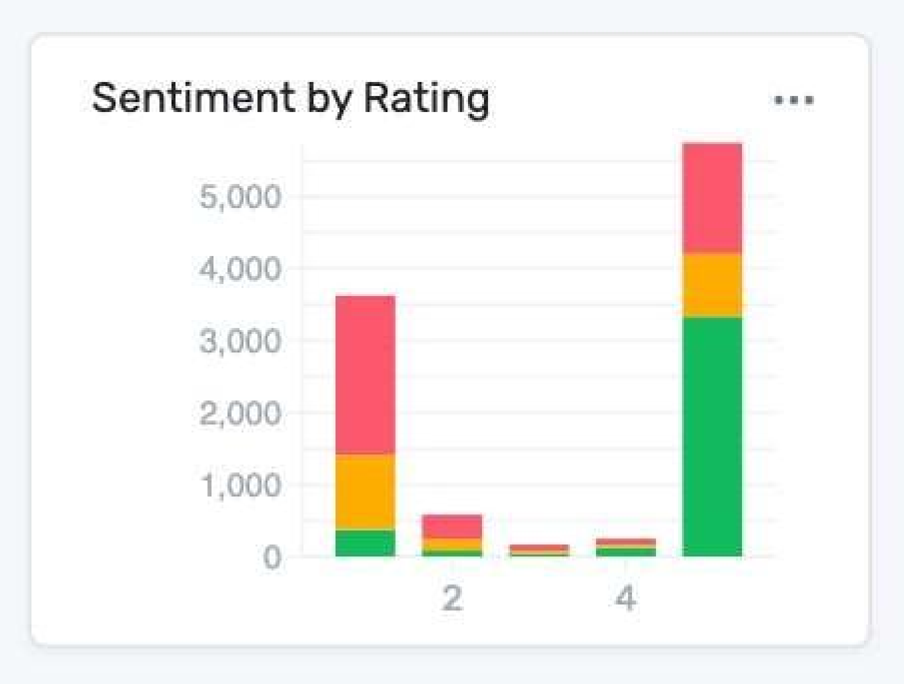


This data visualization sample is classic temporal datavis, a datavis type that tracks results and plots them over a period of time.

This graph expands on our Overall Sentiment data - it tracks the overall proportion of positive, neutral, and negative sentiment in the reviews from 2016 to 2021.

This graph informs the gradual change in the content of their written reviews over this five year period. For instance, negative responses went down from 2019-2020, then jumped back up to previous levels in 2021.

3. Sentiment by rating



Now we jump to something that anchors our text-based sentiment to TrustPilot's earlier results.

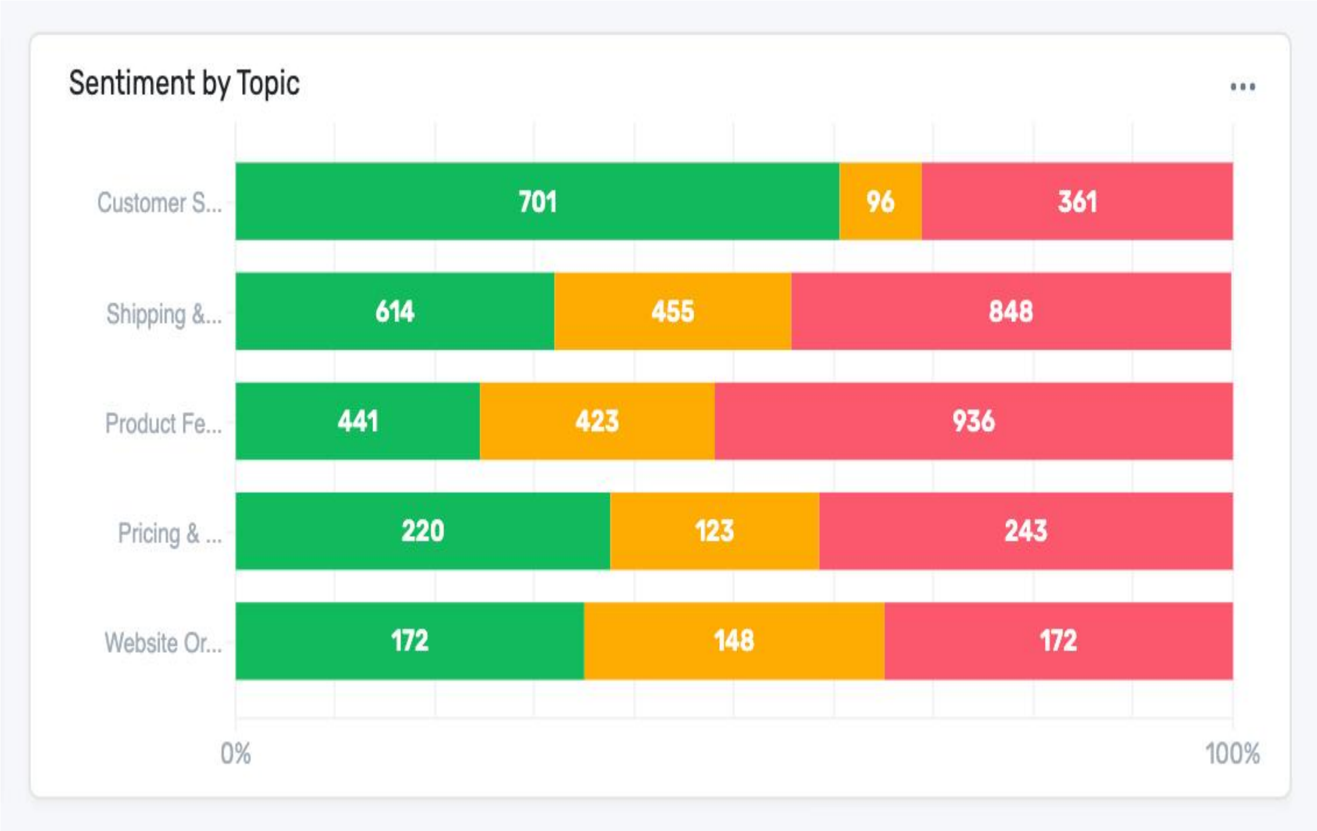
By taking each TrustPilot category from 1-Bad to 5-Excellent, and breaking down the text of the written reviews from the scores you can derive the above graphic.

Looking at the results, and courtesy of taking a deeper look at the reviews via sentiment analysis, we can draw a couple interesting conclusions right off the bat.

1. TrustPilots results aren't useless - the better reviews have higher proportions of positive sentiment and the worse reviews have more negative sentiment. But, all reviews contain a little bit of all types of sentiment - we've learned that our reviews are nuanced and thus likely have even more hidden insight for us!
2. Our reviews are polarized. They skew in amounts towards 5 and 1.

These quick takeaways point us towards goldmines for future analysis. Namely, the positive sentiment sections of negative reviews and the negative section of positive ones, and the 2 - 4 reviews (why do they feel the way they do, how could we improve their scores?).

4. Sentiment by Topic



Finally, we can take a look at Sentiment by Topic to begin to illustrate how sentiment analysis can take us even further into our data.

The above chart applies product-linked text classification in addition to sentiment analysis to pair given sentiment to product/service specific features, this is known as aspect-based sentiment analysis.

This means we can know how our customers feel about what, helping us zero in and fix specific pain points or issues.

These are all great jumping off points designed to visually demonstrate the value of sentiment analysis - but they only scratch the surface of its true power.

Read on for a step-by-step walkthrough of how sentiment analysis works.

How Does Sentiment Analysis Work?

Sentiment analysis, otherwise known as opinion mining, works thanks to natural language processing (NLP) and machine learning algorithms, to automatically determine the emotional tone behind online conversations.

There are different algorithms you can implement in sentiment analysis models, depending on how much data you need to analyze, and how accurate you need your model to be. We'll go over some of these in more detail, below.

Sentiment analysis algorithms fall into one of three buckets:

- **Rule-based:** these systems automatically perform sentiment analysis based on a set of manually crafted rules.
- **Automatic:** systems rely on machine learning techniques to learn from data.
- **Hybrid** systems combine both rule-based and automatic approaches.

Rule-based Approaches

Usually, a rule-based system uses a set of human-crafted rules to help identify subjectivity, polarity, or the subject of an opinion.

These rules may include various NLP techniques developed in computational linguistics, such as:

- *Stemming*, *tokenization*, *part-of-speech tagging* and *parsing*.
- Lexicons (i.e. lists of words and expressions).

Here’s a basic example of how a rule-based system works:

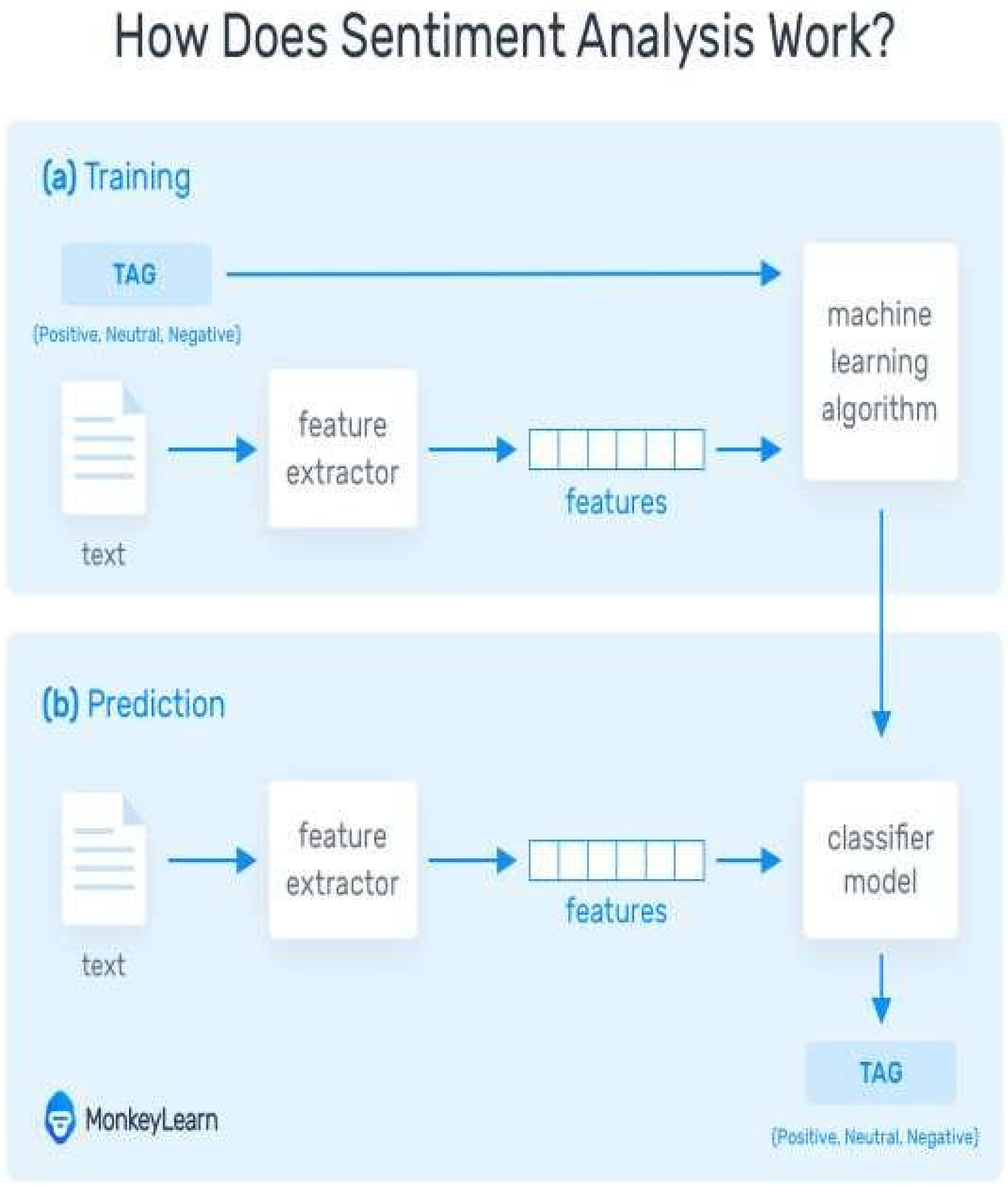
1. Defines two lists of polarized words (e.g. negative words such as *bad*, *worst*, *ugly*, etc and positive words such as *good*, *best*, *beautiful*, etc).
2. Counts the number of positive and negative words that appear in a given text.
3. If the number of positive word appearances is greater than the number of negative word appearances, the system returns a positive sentiment, and vice versa. If the numbers are even, the system will return a neutral sentiment.

Rule-based systems are very naive since they don't take into account how words are combined in a sequence. Of course, more advanced processing techniques can be used, and new rules added to support new expressions and vocabulary. However, adding new rules may affect previous results, and the whole system can get very complex. Since rule-based systems often require fine-tuning and maintenance, they’ll also need regular investments.

Automatic Approaches

Automatic methods, contrary to rule-based systems, don't rely on manually crafted rules, but on machine learning techniques. A sentiment analysis task is usually modeled as a classification problem, whereby a classifier is fed a text and returns a category, e.g. positive, negative, or neutral.

Here’s how a machine learning classifier can be implemented:


How Does Sentiment Analysis Work?
(a) Training
TAG
(Positive, Neutral, Negative)
text
feature extractor
features
machine learning algorithm
(b) Prediction
text
feature extractor
features
classifier model
TAG
(Positive, Neutral, Negative)
MonkeyLearn

The Training and Prediction Processes

In the training process (a), our model learns to associate a particular input (i.e. a text) to the corresponding output (tag) based on the test samples used for training. The feature extractor transfers the text input into a feature vector. Pairs of feature vectors and tags (e.g. *positive*, *negative*, or *neutral*) are fed into the machine learning algorithm to generate a model.

In the prediction process (b), the feature extractor is used to transform unseen text inputs into feature vectors. These feature vectors are then fed into the model, which generates predicted tags (again, *positive*, *negative*, or *neutral*).

Feature Extraction from Text

The first step in a machine learning text classifier is to transform the text extraction or text vectorization, and the classical approach has been bag-of-words or bag-of-ngrams with their frequency.

More recently, new feature extraction techniques have been applied based on word embeddings (also known as *word vectors*). This kind of representations makes it possible for words with similar meaning to have a similar representation, which can improve the performance of classifiers.

Classification Algorithms

The classification step usually involves a statistical model like Naïve Bayes, Logistic Regression, Support Vector Machines, or Neural Networks:

- Naïve Bayes: a family of probabilistic algorithms that uses Bayes's Theorem to predict the category of a text.
- Linear Regression: a very well-known algorithm in statistics used to predict some value (Y) given a set of features (X).
- Support Vector Machines: a non-probabilistic model which uses a representation of text examples as points in a multidimensional space. Examples of different categories (sentiments) are mapped to distinct regions within that space. Then, new texts are assigned a category based on similarities with existing texts and the regions they're mapped to.
- Deep Learning: a diverse set of algorithms that attempt to mimic the human brain, by employing artificial neural networks to process data.

Hybrid Approaches

Hybrid systems combine the desirable elements of rule-based and automatic techniques into one system. One huge benefit of these systems is that results are often more accurate.

Sentiment Analysis Challenges

Sentiment analysis is one of the hardest tasks in natural language processing because even humans struggle to analyze sentiments accurately.

Data scientists are getting better at creating more accurate sentiment classifiers, but there's still a long way to go. Let's take a closer look at some of the main challenges of machine-based sentiment analysis:

Subjectivity and Tone

There are two types of text: subjective and objective. Objective texts do not contain explicit sentiments, whereas subjective texts do. Say, for example, you intend to analyze the sentiment of the following two texts:

The package is nice.

The package is red.

Most people would say that sentiment is positive for the first one and neutral for the second one, right? All *predicates* (adjectives, verbs, and some nouns) should not be treated the same with respect to how they create sentiment. In the examples above, *nice* is more *subjective* than *red*.

Context and Polarity

All utterances are uttered at some point in time, in some place, by and to some people, you get the point. All utterances are uttered in context. Analyzing sentiment without context gets pretty difficult. However, machines cannot learn about contexts if they are not mentioned explicitly. One of the problems that arise from context is changes in polarity. Look at the following responses to a survey:

Everything about it.

Absolutely nothing!

Imagine the responses above come from answers to the question *What did you like about the event?* The first response would be positive and the second one would be negative, right? Now, imagine the responses come from answers to the question *What did you DISlike about the event?* The negative in the question will make sentiment analysis change altogether.

A good deal of preprocessing or postprocessing will be needed if we are to take into account at least part of the context in which texts were produced. However, how to preprocess or postprocess data in order to capture the bits of context that will help analyze sentiment is not straightforward.

Irony and Sarcasm

When it comes to *irony* and *sarcasm*, people express their negative sentiments using positive words, which can be difficult for machines to detect without having a thorough understanding of the context of the situation in which a feeling was expressed.

For example, look at some possible answers to the question, *Did you enjoy your shopping experience with us?*

Yeah, sure. So smooth!

Not one, but many!

What sentiment would you assign to the responses above? The first response with an exclamation mark could be negative, right? The problem is there is no textual cue that will help a machine learn, or at least question that sentiment since *yeah* and *sure* often belong to positive or neutral texts.

How about the second response? In this context, sentiment is positive, but we're sure you can come up with many different contexts in which the same response can express negative sentiment.

Comparisons

How to treat comparisons in sentiment analysis is another challenge worth tackling. Look at the texts below:

This product is second to none.

This is better than older tools.

This is better than nothing.

The first comparison doesn't need any contextual clues to be classified correctly. It's clear that it's positive.

The second and third texts are a little more difficult to classify, though. Would you classify them as *neutral*, *positive*, or even *negative*? Once again, context can make a difference. For example, if the *'older tools'* in the second text were considered useless, then the second text is pretty similar to the third text.

Emojis

There are two types of emojis according to Guibon et al.. *Western emojis* (e.g. :D) are encoded in only one or two characters, whereas *Eastern emojis* (e.g. ¯ \ (ツ) / ¯) are a longer combination of characters of a vertical nature. Emojis play an important role in the sentiment of texts, particularly in tweets.

You'll need to pay special attention to character-level, as well as word-level, when performing sentiment analysis on tweets. A lot of preprocessing might also be needed. For example, you might want to preprocess social media content and transform both Western and Eastern emojis into tokens and whitelist them (i.e. always take them as a feature for classification purposes) in order to help improve sentiment analysis performance.

Here's a quite comprehensive list of emojis and their unicode characters that may come in handy when preprocessing.

Defining Neutral

Defining what we mean by *neutral* is another challenge to tackle in order to perform accurate sentiment analysis. As in all classification problems, defining your categories -and, in this case, the *neutral* tag- is one of the most important parts of the problem. What *you* mean by *neutral*, *positive*, or *negative* does matter when you train sentiment analysis models. Since tagging data requires that tagging criteria be consistent, a good definition of the problem is a must.

Here are some ideas to help you identify and define *neutral* texts:

1. **Objective texts**. So called *objective* texts do not contain explicit sentiments, so you should include those texts into the neutral category.
2. **Irrelevant information**. If you haven't preprocessed your data to filter out irrelevant information, you can tag it neutral. However, be careful! Only do this if you know how this could affect overall performance. Sometimes, you will be adding noise to your classifier and performance could get worse.
3. **Texts containing wishes**. Some wishes like, *I wish the product had more integrations* are generally neutral. However, those including comparisons like, *I wish the product were better* are pretty difficult to categorize

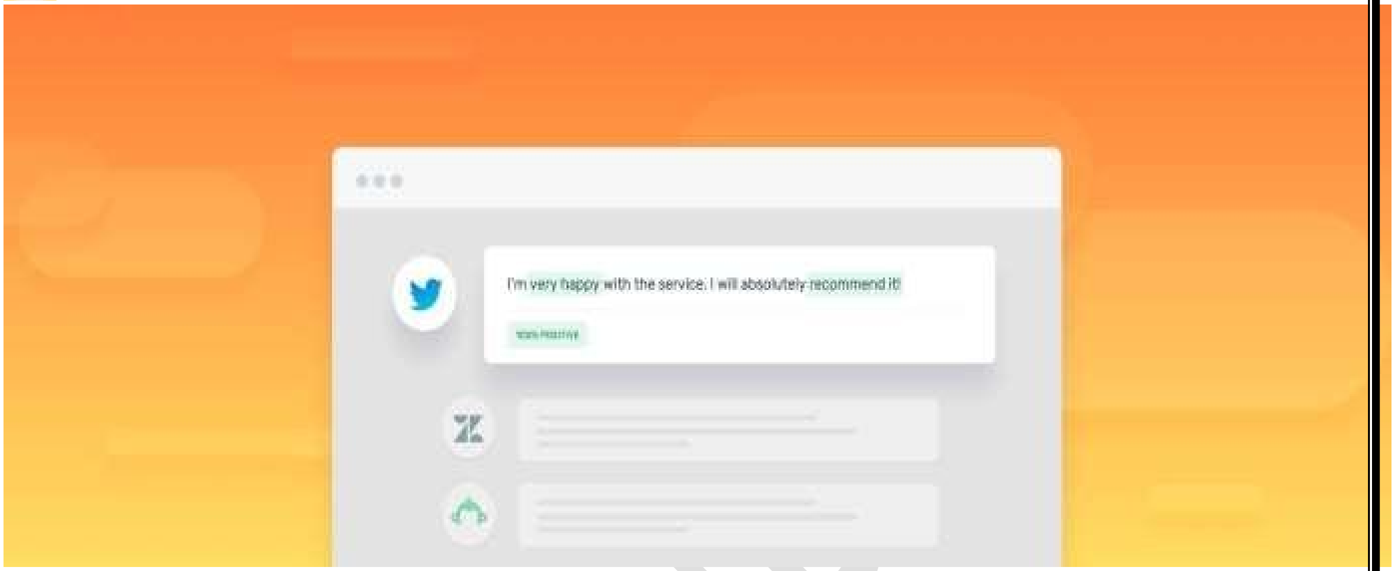
Human Annotator Accuracy

Sentiment analysis is a tremendously difficult task even for humans. On average, inter-annotator agreement (a measure of how well two (or more) human labelers can make the same annotation decision) is pretty low when it comes to sentiment analysis. And since machines learn from labeled data, sentiment analysis classifiers might not be as precise as other types of classifiers.

Still, sentiment analysis is worth the effort, even if your sentiment analysis predictions are wrong from time to time. By using MonkeyLearn's sentiment analysis model, you can expect correct predictions about 70-80% of the time you submit your texts for classification.

If you are new to sentiment analysis, then you'll quickly notice improvements. For typical use cases, such as ticket routing, brand monitoring, and VoC analysis, you'll save a lot of time and money on tedious manual tasks.

Sentiment Analysis Use Cases & Applications



I'm very happy with the service. I will absolutely recommend it!

The applications of sentiment analysis are endless and can be applied to any industry, from finance and retail to hospitality and technology. Below, we've listed some of the most popular ways that sentiment analysis is being used in business:

1. Social Media Monitoring
2. Brand Monitoring
3. Voice of customer (VoC)
4. Customer Service
5. Market Research

Social Media Monitoring

Sentiment analysis is used in social media monitoring, allowing businesses to gain insights about how customers feel about certain topics, and detect urgent issues in real time before they spiral out of control.

On the fateful evening of April 9th, 2017, United Airlines forcibly removed a passenger from an overbooked flight. The nightmare-ish incident was filmed by other passengers on their smartphones and posted immediately. One of the videos, posted to Facebook, was shared more than 87,000 times and viewed 6.8 million times by 6pm on Monday, just 24 hours later.

The fiasco was only magnified by the company's dismissive response. On Monday afternoon, United's CEO tweeted a statement apologizing for "having to re-accommodate customers."

This is exactly the kind of PR catastrophe you can avoid with sentiment analysis. It's an example of why it's important to care, not only about if people are talking about your brand, but how they're talking about it. More mentions don't equal positive mentions.

Brands of all shapes and sizes have meaningful interactions with customers, leads, even their competition, all across social media. By monitoring these conversations you can understand customer sentiment in real time and over time, so you can detect disgruntled customers immediately and respond as soon as possible.

Most marketing departments are already tuned into online mentions as far as volume – they measure more chatter as more brand awareness. But businesses need to look beyond the numbers for deeper insights.

Brand Monitoring

Not only do brands have a wealth of information available on social media, but across the internet, on news sites, blogs, forums, product reviews, and more. Again, we can look at not just the volume of mentions, but the individual and overall quality of those mentions.

In our United Airlines example, for instance, the flare-up started on the social media accounts of just a few passengers. Within hours, it was picked up by news sites and spread like wildfire across the US, then to China and Vietnam, as United was accused of racial profiling against a passenger of Chinese-Vietnamese descent. In China, the incident became the number one trending topic on Weibo, a microblogging site with almost 500 million users.

And again, this is all happening within mere hours of the incident.

Brand monitoring offers a wealth of insights from conversations happening about your brand from all over the internet. Analyze news articles, blogs, forums, and more to gauge brand sentiment, and target certain demographics or regions, as desired. Automatically categorize the urgency of all brand mentions and route them instantly to designated team members.

Get an understanding of customer feelings and opinions, beyond mere numbers and statistics. Understand how your brand image evolves over time, and compare it to that of your competition. You can tune into a specific point in time to follow product releases, marketing campaigns, IPO filings, etc., and compare them to past events.

Real-time sentiment analysis allows you to identify potential PR crises and take immediate action before they become serious issues. Or identify positive comments and respond directly, to use them to your benefit.

Example: Expedia Canada

Around Christmas time, Expedia Canada ran a classic "escape winter" marketing campaign. All was well, except for the screeching violin they chose as background music. Understandably, people took to social media, blogs, and forums. Expedia noticed right away and removed the ad.

Then, they created a series of follow-up spin-off videos: one showed the original actor smashing the violin; another invited a real negative Twitter user to rip the violin out of the actor's hands on screen. Though their original campaign was a flop, Expedia were able to redeem themselves by listening to their customers and responding.

Sentiment analysis allows you to automatically monitor all chatter around your brand and detect and address this type of potentially-explosive scenario while you still have time to defuse it.

Voice of Customer (VoC)

Social media and brand monitoring offer us immediate, unfiltered, and invaluable information on customer sentiment, but you can also put this analysis to work on surveys and customer support interactions.

Net Promoter Score (NPS) surveys are one of the most popular ways for businesses to gain feedback with the simple question: Would you recommend this company, product, and/or service to a friend or family member? These result in a single score on a number scale.

Businesses use these scores to identify customers as promoters, passives, or detractors. The goal is to identify overall customer experience, and find ways to elevate all customers to "promoter" level, where they, theoretically, will buy more, stay longer, and refer other customers.

Numerical (quantitative) survey data is easily aggregated and assessed. But the next question in NPS surveys, asking why survey participants left the score they did, seeks open-ended responses, or qualitative data.

Open-ended survey responses were previously much more difficult to analyze, but with sentiment analysis these texts can be classified into positive and negative (and everywhere in between) offering further insights into the Voice of Customer (VoC).

Sentiment analysis can be used on any kind of survey – quantitative and qualitative – and on customer support interactions, to understand the emotions and opinions of your customers. Tracking customer sentiment over time adds depth to help understand why NPS scores or sentiment toward individual aspects of your business may have changed.

You can use it on incoming surveys and support tickets to detect customers who are 'strongly negative' and target them immediately to improve their service. Zero in on certain demographics to understand what works best and how you can improve.

Real-time analysis allows you to see shifts in VoC right away and understand the nuances of the customer experience over time beyond statistics and percentages.

Discover how we analyzed the sentiment of thousands of Facebook reviews, and transformed them into actionable insights.

Example: McKinsey City Voices project

In Brazil, federal public spending rose by 156% from 2007 to 2015, while satisfaction with public services steadily decreased. Unhappy with this counterproductive progress, the Urban Planning Department recruited McKinsey to help them focus on user experience, or "citizen journeys," when delivering services. This citizen-centric style of governance has led to the rise of what we call Smart Cities.

McKinsey developed a tool called City Voices, which conducts citizen surveys across more than 150 metrics, and then runs sentiment analysis to help leaders understand how constituents live and what they need, in order to better inform public policy. By using this tool, the Brazilian government was able to uncover the most urgent needs – a safer bus system, for instance – and improve them first.

If this can be successful on a national scale, imagine what it can do for your company.

Customer Service

We already looked at how we can use sentiment analysis in terms of the broader VoC, so now we'll dial in on customer service teams.

We all know the drill: stellar customer experiences means a higher rate of returning customers. Leading companies know that how they deliver is just as, if not more, important as what they deliver. Customers expect their experience with companies to be immediate, intuitive, personal, and hassle-free. If not, they'll leave and do business elsewhere. Did you know that one in three customers will leave a brand after just one bad experience?

You can use sentiment analysis and text classification to automatically organize incoming support queries by topic and urgency to route them to the correct department and make sure the most urgent are handled right away.

Analyze customer support interactions to ensure your employees are following appropriate protocol. Increase efficiency, so customers aren't left waiting for support. Decrease churn rates; after all it's less hassle to keep customers than acquire new ones.

Market Research

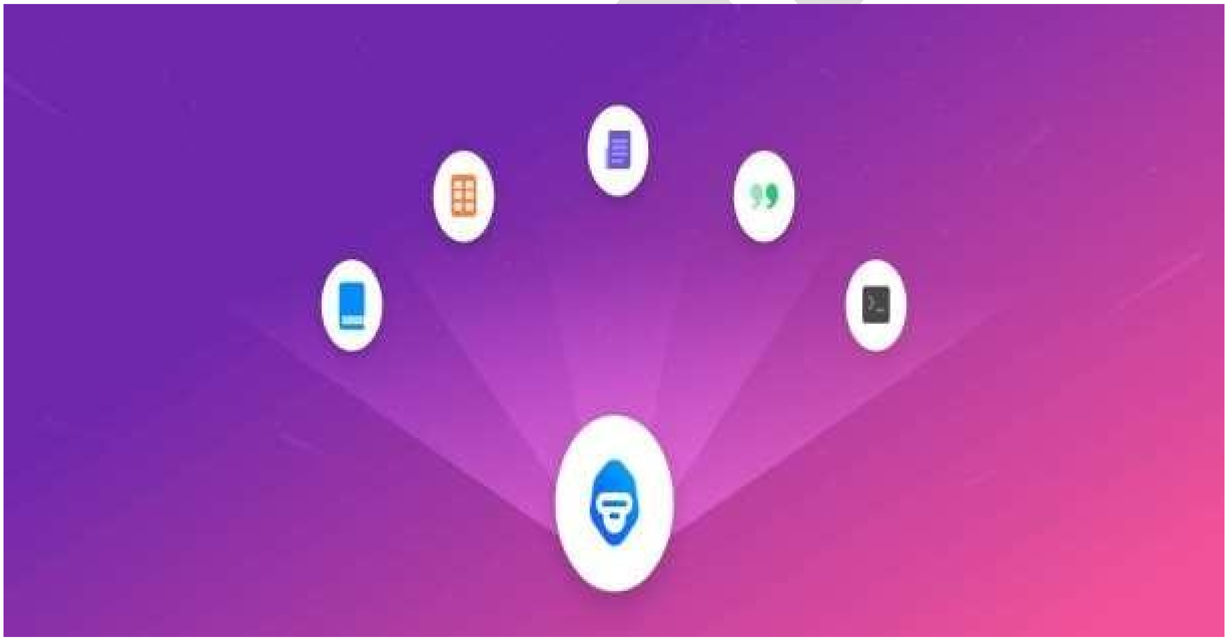
Sentiment analysis empowers all kinds of market research and competitive analysis. Whether you're exploring a new market, anticipating future trends, or seeking an edge on the competition, sentiment analysis can make all the difference.

You can analyze online reviews of your products and compare them to your competition. Maybe your competitor released a new product that landed as a flop. Find out what aspects of the product performed most negatively and use it to your advantage.

Follow your brand and your competition in real time on social media. Locate new markets where your brand is likely to succeed. Uncover trends just as they emerge, or follow long-term market leanings through analysis of formal market reports and business journals.

You'll tap into new sources of information and be able to quantify otherwise qualitative information. With social data analysis you can fill in gaps where public data is scarce, like emerging markets.

Sentiment Analysis Tools & Tutorials



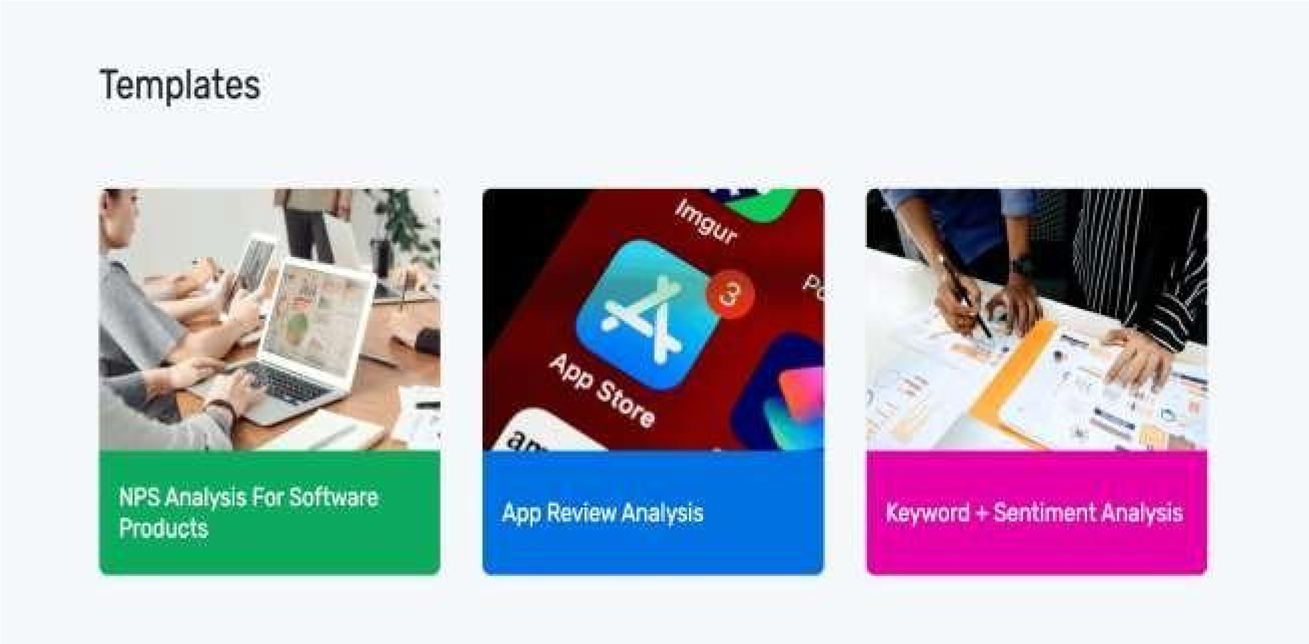
Sentiment analysis is a vast topic, and it can be intimidating to get started. Luckily, there are many useful resources, from helpful tutorials to all kinds of free online tools, to help you take your first steps.

Free Online Sentiment Analysis Tools

A good start to your journey is to simply play around with a sentiment analysis tool. A little first-hand experience will help you understand how it works

Next, to take your sentiment analysis further, you'll want to try out MonkeyLearn's sentiment analysis and keyword template. First, you'll need sign up, then walk through the following steps:

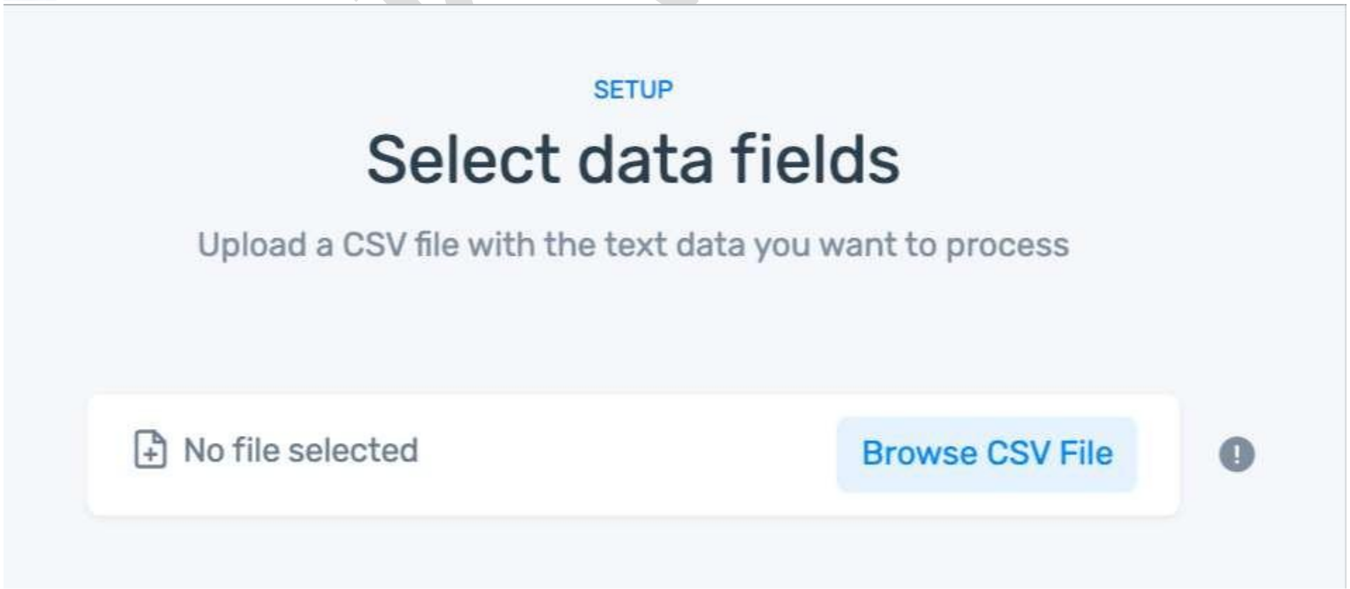
1. Choose Keyword + Sentiment Analysis template



Templates

NPS Analysis For Software Products
App Review Analysis
Keyword + Sentiment Analysis

2. Upload your data



SETUP

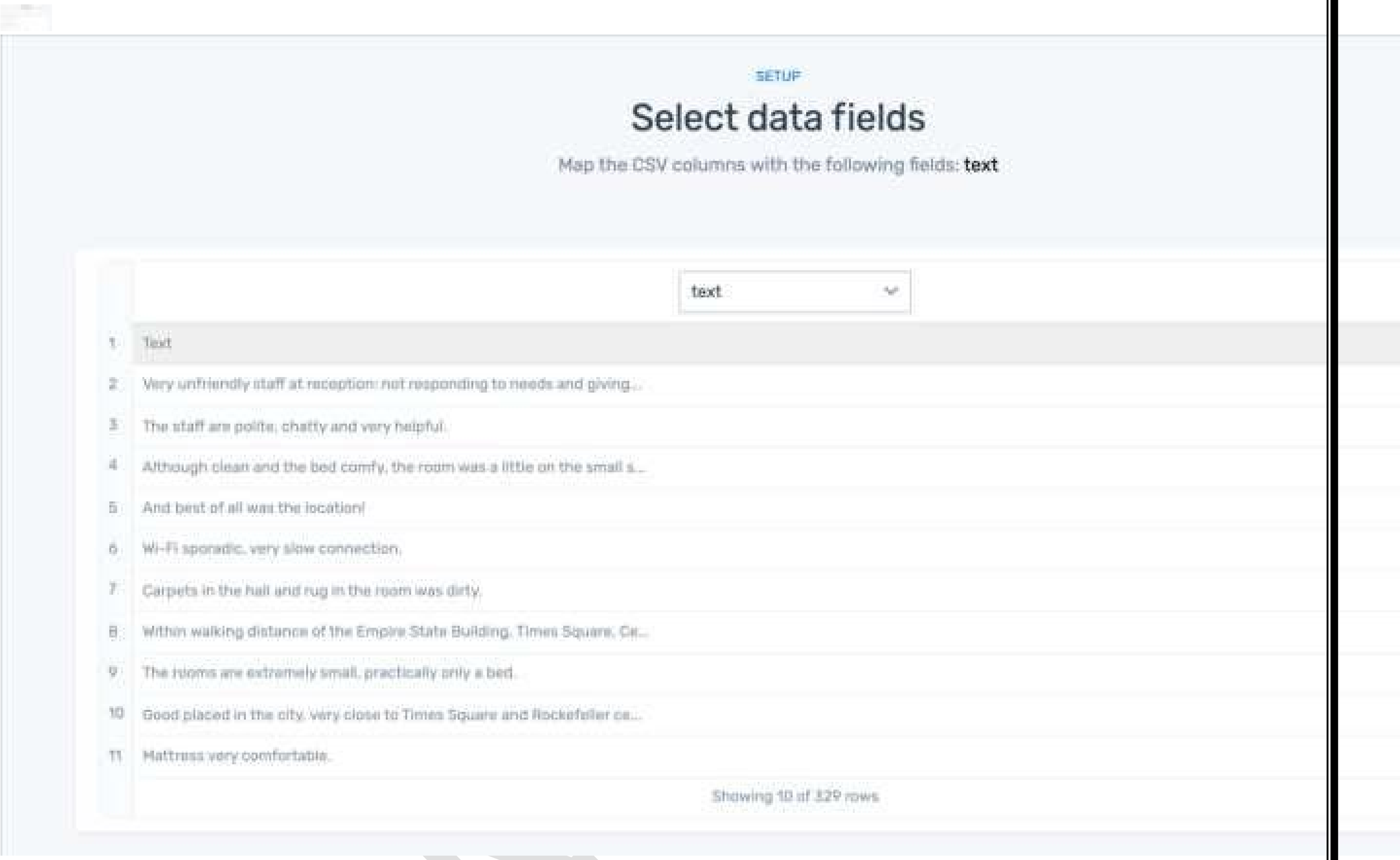
Select data fields

Upload a CSV file with the text data you want to process

If you don't have a CSV, you can use our sample dataset.

3. Match the CSV columns to the dashboard fields

In this template, there is only one field: text. If you have more than one column in your dataset, choose the column that has the text you would like to analyze.



SETUP

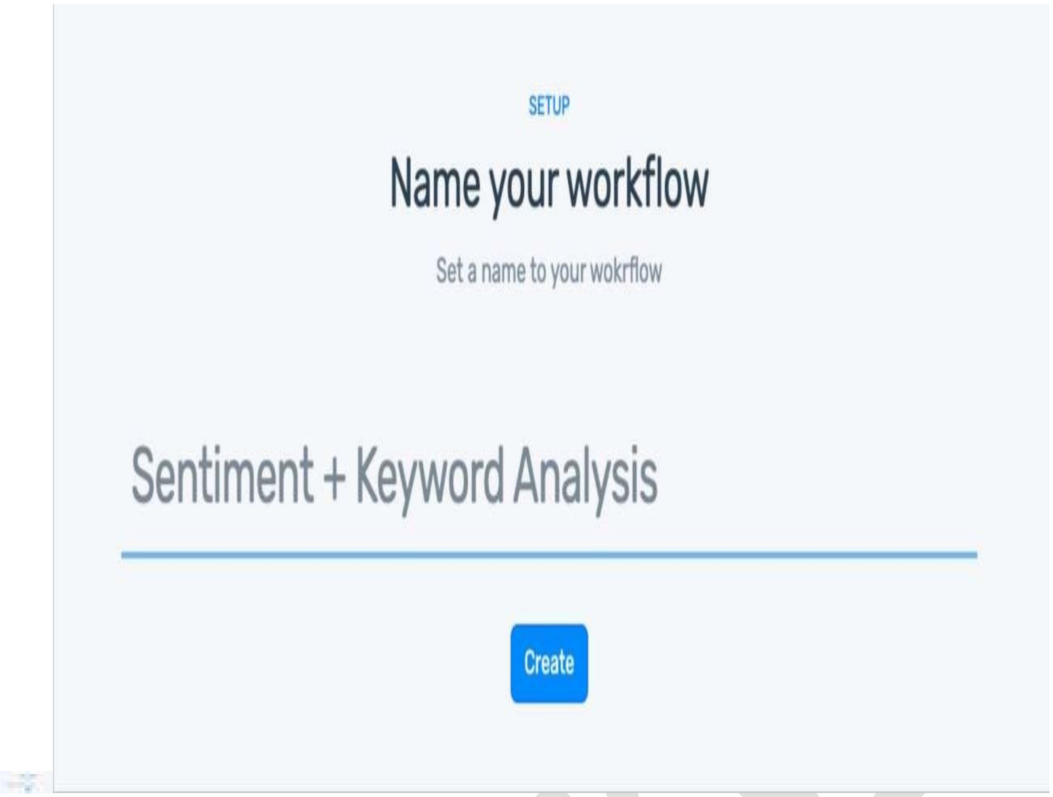
Select data fields

Map the CSV columns with the following fields: text

	text
1	Text
2	Very unfriendly staff at reception: not responding to needs and giving...
3	The staff are polite, chatty and very helpful.
4	Although clean and the bed comfy, the room was a little on the small s...
5	And best of all was the location!
6	Wi-Fi sporadic, very slow connection.
7	Carpets in the hall and rug in the room was dirty.
8	Within walking distance of the Empire State Building, Times Square, Ce...
9	The rooms are extremely small, practically only a bed.
10	Good placed in the city, very close to Times Square and Rockefeller ce...
11	Mattress very comfortable.

Showing 10 of 329 rows

4. Name your workflow



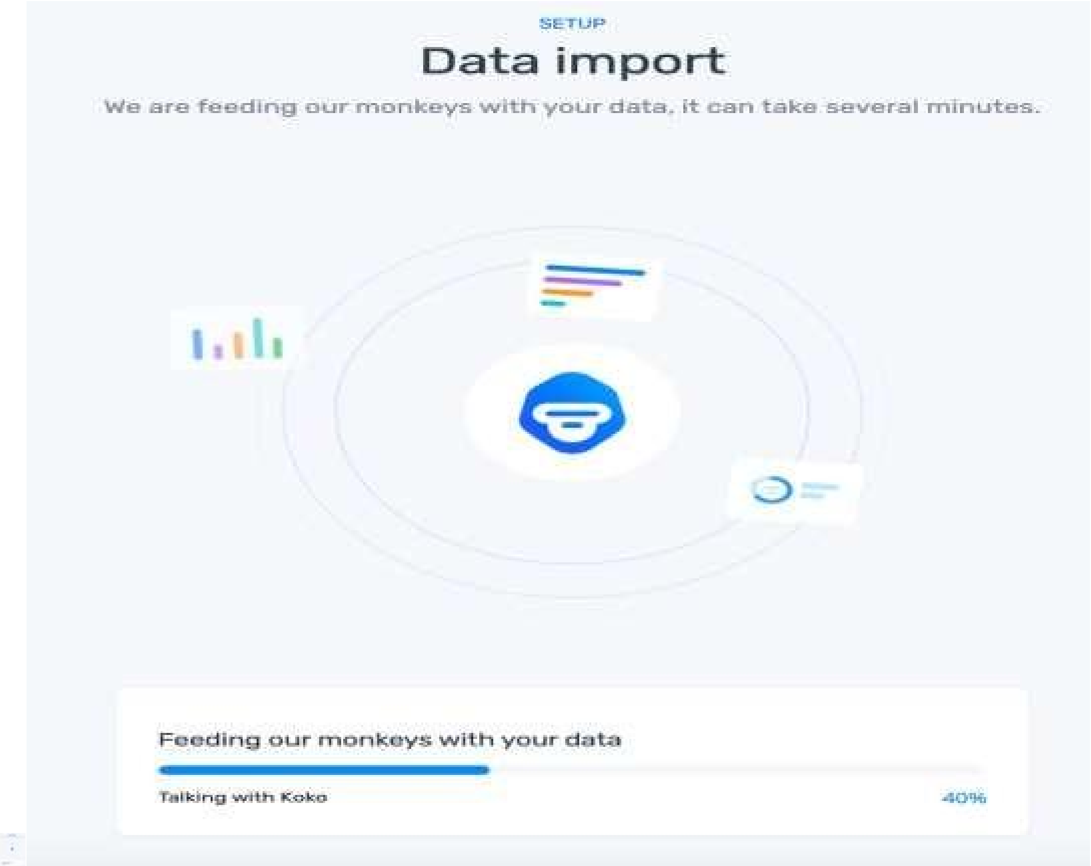
SETUP

Name your workflow

Set a name to your wokrflow

Sentiment + Keyword Analysis

5. Wait for your data to import



SETUP

Data import

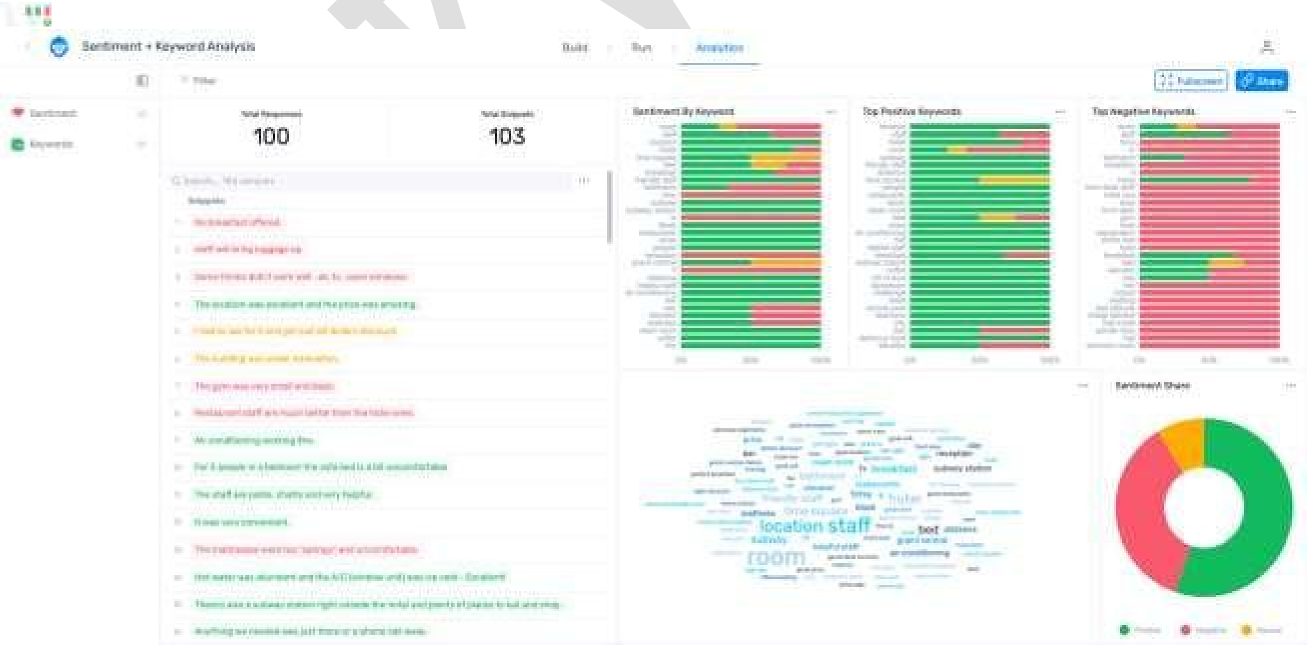
We are feeding our monkeys with your data, it can take several minutes.

Feeding our monkeys with your data

Talking with Koko

40%

6. Explore your dashboard!



No breakfast offered.

Staff will bring luggage up

[illegible]

The location was excellent and the price was amazing

[illegible]

The building was under renovation.

The gym was very small and basic

Restaurant staff are much better than the hotel ones

Air conditioning working fine

For 3 people in a bedroom the sofa bed is a bit uncomfortable

The staff are polite, chatty and very helpful

It was very convenient.

[illegible]

Hot water was abundant and the A/C [illegible]

There was a subway station right outside the hotel and plenty of places to eat and shop

Anything we needed was just there or a phone call away.

You can:

- Filter by sentiment or keyword.
- Share via email with other coworkers.

Open Source vs SaaS (Software as a Service) Sentiment Analysis Tools

When it comes to sentiment analysis (and text analysis in general), you have two choices: build your own solution or buy a tool.

Open source libraries in languages like Python and Java are particularly well positioned to build your own sentiment analysis solution because their communities lean more heavily toward data science, like natural language processing and deep learning for sentiment analysis. But you'll need a team of data scientists and engineers on board, huge upfront investments, and time to spare.

SaaS tools offer the option to implement pre-trained sentiment analysis models immediately or custom-train your own, often in just a few steps. These tools are recommended if you don't have a data science or engineering team on board, since they can be implemented with little or no code and can save months of work and money (upwards of $100,000).

If you're still convinced that you need to build your own sentiment analysis solution, check out these tools and tutorials in various programming languages:

Sentiment Analysis Python

- Scikit-learn is the go-to library for machine learning and has useful tools for text vectorization. Training a classifier on top of vectorizations, like frequency or tf-idf text vectorizers is quite straightforward. Scikit-learn has implementations for Support Vector Machines, Naïve Bayes, and Logistic Regression, among others.
- NLTK has been the traditional NLP library for Python. It has an active community and offers the possibility to train machine learning classifiers.
- SpaCy is an NLP library with a growing community. Like NLTK, it provides a strong set of low-level functions for NLP and support for training text classifiers.
- TensorFlow, developed by Google, provides a low-level set of tools to build and train neural networks. There's also support for text vectorization, both on traditional word frequency and on more advanced through-word embeddings.
- Keras provides useful abstractions to work with multiple neural network types, like recurrent neural networks (RNNs) and convolutional neural networks (CNNs) and easily stack layers of neurons. Keras can be run on top of Tensorflow or Theano. It also provides useful tools for text classification.
- PyTorch is a recent deep learning framework backed by some prestigious organizations like Facebook, Twitter, Nvidia, Salesforce, Stanford University, University of Oxford, and Uber. It has quickly developed a strong community.

Sentiment Analysis Javascript

Java is another programming language with a strong community around data science with remarkable data science libraries for NLP.

- OpenNLP: a toolkit that supports the most common NLP tasks, such as tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, language detection and coreference resolution.
- Stanford CoreNLP: a Java suite of core NLP tools provided by The Stanford NLP Group.
- Lingpipe: a Java toolkit for processing text using computational linguistics. LingPipe is often used for text classification and entity extraction.
- Weka: a set of tools created by The University of Waikato for data pre-processing, classification, regression, clustering, association rules, and visualization.

Sentiment Analysis Research & Courses

After learning the basics of sentiment analysis, and understanding how it can help you, you might want to delve further into the topic:

Sentiment Analysis Papers

The literature around sentiment analysis is massive; there are more than 55,700 scholarly articles, papers, theses, books, and abstracts out there.

The following are the most frequently cited and read papers in the sentiment analysis community in general:

- Opinion mining and sentiment analysis (Pang and Lee, 2008)
- Recognizing contextual polarity in phrase-level sentiment analysis (Wilson, Wiebe and Hoffmann, 2005).
- A survey of opinion mining and sentiment analysis (Liu and Zhang, 2012)
- Sentiment analysis and opinion mining (Liu, 2012)
- How to Perform Text Mining with Sentiment Analysis

Sentiment Analysis Books

Bing Liu is a thought leader in the field of machine learning and has written a book about sentiment analysis and opinion mining.

Useful for those starting research on sentiment analysis, Liu does a wonderful job of explaining sentiment analysis in a way that is highly technical, yet understandable. In the book, he covers different aspects of sentiment analysis including applications, research, sentiment classification using supervised and unsupervised learning, sentence subjectivity, aspect-based sentiment analysis, and more.

For those who want to learn about deep-learning based approaches for sentiment analysis, a relatively new and fast-growing research area, take a look at Deep-Learning Based Approaches for Sentiment Analysis.

Sentiment Analysis Courses and Lectures

Another good way to go deeper with sentiment analysis is mastering your knowledge and skills in natural language processing (NLP), the computer science field that focuses on understanding 'human' language.

By combining machine learning, computational linguistics, and computer science, NLP allows a machine to understand natural language including people's sentiments, evaluations, attitudes, and emotions from written language.

There are a large number of courses, lectures, and resources available online, but the essential NLP course is the Stanford Coursera course by Dan Jurafsky and Christopher Manning. By taking this course, you will get a step-by-step introduction to the field by two of the most reputable names in the NLP community.

If you want a more hands-on course, you should enroll in the Data Science: Natural Language Processing (NLP) in Python on Udemy. This course gives you a good introduction to NLP and what it can do, but it will also make you build different projects in Python, including a spam detector, a sentiment analyzer, and an article spinner. Most of the lectures are really short (~5 minutes) and the course strikes the right balance between practical and theoretical content.

Sentiment Analysis Datasets

The key part for mastering sentiment analysis is working on different datasets and experimenting with different approaches. First, you'll need to get your hands on data and procure a dataset which you will use to carry out your experiments.

The following are some of our favorite sentiment analysis datasets for experimenting with sentiment analysis and a machine learning approach. They're open and free to download:

- Product reviews: this dataset consists of a few million Amazon customer reviews with star ratings, super useful for training a sentiment analysis model.
- Restaurant reviews: this dataset consists of 5,2 million Yelp reviews with star ratings.
- Movie reviews: this dataset consists of 1,000 positive and 1,000 negative processed reviews. It also provides 5,331 positive and 5,331 negative processed sentences / snippets.
- Fine food reviews: this dataset consists of ~500,000 food reviews from Amazon. It includes product and user information, ratings, and a plain text version of every review.
- Twitter airline sentiment on Kaggle: this dataset consists of ~15,000 labeled tweets (positive, neutral, and negative) about airlines.
- First GOP Debate Twitter Sentiment: this dataset consists of ~14,000 labeled tweets (positive, neutral, and negative) about the first GOP debate in 2016.

If you are interested in rule-based approach, the following is a varied list of sentiment analysis lexicons that will come in handy. These lexicons provide a set of dictionaries of words with labels specifying their sentiments across different domains. The following lexicons are really useful to identify the sentiment of texts:

- Sentiment Lexicons for 81 Languages: this dataset contains both positive and negative sentiment lexicons for 81 languages.
- SentiWordNet: this dataset contains about 29,000 words with a sentiment score between 0 and 1.
- Opinion Lexicon for Sentiment Analysis: this dataset provides a list of 4,782 negative words and 2,005 positive words in English.
- Wordstat Sentiment Dictionary: this dataset includes ~4800 positive and ~9000 negative words.
- Emoticon Sentiment Lexicon: this dataset contains a list of 477 emoticons labeled as positive, neutral, or negative.

Parting words

Sentiment analysis can be applied to countless aspects of business, from brand monitoring and product analytics, to customer service and market research. By incorporating it into their existing systems and analytics, leading brands (not to mention entire cities) are able to work faster, with more accuracy, toward more useful ends.

Sentiment analysis has moved beyond merely an interesting, high-tech whim, and will soon become an indispensable tool for all companies of the modern age. Ultimately, sentiment analysis enables us to glean new insights, better understand our customers, and empower our own teams more effectively so that they do better and more productive work.

A transformer is **a deep learning model that adopts the mechanism of self-attention, differentially weighting the significance of each part of the input data**. It is used primarily in the fields of natural language processing (NLP) and computer vision (CV).

Ethics in AI:

AI ethics is a set of guidelines that advise on the design and outcomes of artificial intelligence. Human beings come with all sorts of cognitive biases, such as recency and confirmation bias, and those inherent biases are exhibited in our behaviours and subsequently, our data.

1. Respect for Persons: This principle recognizes the autonomy of individuals and upholds an expectation for researchers to protect individuals with diminished autonomy, which could be due to a variety of circumstances such as illness, a mental disability, age restrictions.

2. Beneficence: This principle takes a page out of healthcare ethics, where doctors take an oath to "do no harm."

3. Justice: This principle deals with issues, such as fairness and equality. Who should reap the benefits of experimentation and machine learning? The Belmont Report offers five ways to distribute burdens and benefits, which are by:

- **Equal share**
- **Societal contribution**
- **Individual need**
- **Merit**
- **Individual effort**

What are ethics in AI?

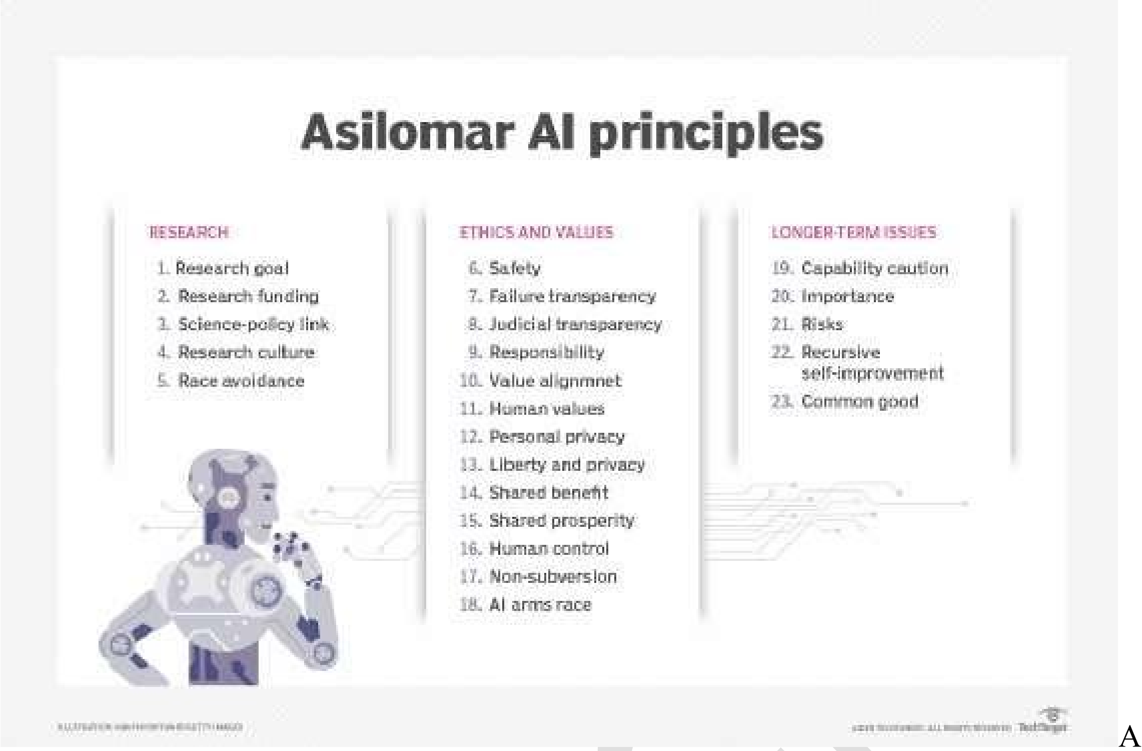
AI ethics is a system of moral principles and techniques intended to inform the development and responsible use of artificial intelligence technology. As AI has become integral to products and services, organizations are starting to develop AI codes of ethics.

An AI code of ethics, also called an AI value platform, is a policy statement that formally defines the role of artificial intelligence as it applies to the continued development of the human race. The purpose of an AI code of ethics is to provide stakeholders with guidance when faced with an ethical decision regarding the use of artificial intelligence.

Isaac Asimov, the science fiction writer, foresaw the potential dangers of autonomous AI agents long before their development and created The Three Laws of Robotics as a means of limiting those risks. In Asimov's code of ethics, the first law forbids robots from actively harming humans or allowing harm to come to humans by refusing to act. The second law orders robots to obey humans unless the orders are not in accordance with the first law. The third law orders robots to protect themselves insofar as doing so is in accordance with the first two laws.

The rapid advancement of AI in the past five to 10 years has spurred groups of experts to develop safeguards for protecting against the risk of AI to humans. One such group is the nonprofit institute founded by MIT cosmologist Max Tegmark, Skype co-founder Jaan Tallinn and DeepMind research scientist Victoria Krakovna. The institute worked with AI researchers and developers as well as scholars from many disciplines to create the 23 guidelines now referred to as the Asilomar AI Principles.

Kelly Combs, director at Digital Lighthouse, KPMG said that when developing an AI code of ethics "it's imperative to include clear guidelines on how the technology will be deployed and continuously monitored." These policies should mandate measures that guard against unintended bias in machine-learning algorithms, continuously detect drift in data and algorithms, and track both the provenance of data and the identity of those who train algorithms.



Asilomar AI principles

RESEARCH

1. Research goal
2. Research funding
3. Science-policy link
4. Research culture
5. Race avoidance

ETHICS AND VALUES

6. Safety
7. Failure transparency
8. Judicial transparency
9. Responsibility
10. Value alignmnet
11. Human values
12. Personal privacy
13. Liberty and privacy
14. Shared benefit
15. Shared prosperity
16. Human control
17. Non-subversion
18. AI arms race

LONGER-TERM ISSUES

19. Capability caution
20. Importance
21. Risks
22. Recursive self-improvement
23. Common good

AI experts and scholars from many disciplines created 23 guidelines now referred to as the Asilomar AI Principles.

Why are AI ethics are important?

AI is a technology designed by humans to replicate, augment or replace human intelligence. These tools typically rely on large volumes of various types of data to develop insights. Poorly designed projects built on data that is faulty, inadequate or biased can have unintended, potentially harmful, consequences. Moreover, the rapid advancement in algorithmic systems means that in some cases it is not clear to us how the AI reached its conclusions, so we are essentially relying on systems we can't explain to make decisions that could affect society.

An AI ethics framework is important because it shines a light on the risks and benefits of AI tools and establishes guidelines for its responsible use. Coming up with a system of moral tenets and techniques for using AI responsibly requires the industry and interested parties to examine major social issues and ultimately the question of what makes us human.

What are the ethical challenges of AI?

Enterprises face several ethical challenges in their use of AI technology.

- Explainability. When AI systems go awry, teams need to be able to trace through a complex chain of algorithmic systems and data processes to find out why. Organizations using AI should be able to explain the source data, resulting data, what their algorithms do and why they are doing that. "AI needs to have a strong degree of traceability to ensure that if harms arise, they can be traced back to the cause," said Adam Wisniewski, CTO and co-founder of AI Clearing.

- Responsibility. Society is still sorting out responsibility when decisions made by AI systems have catastrophic consequences, including loss of capital, health or life. Responsibility for the consequences of AI-based decisions needs to be sorted out in a process that includes lawyers, regulators and citizens. One challenge is finding the appropriate balance in cases where an AI system may be safer than the human activity it is duplicating but still causes problems, such as weighing the merits of autonomous driving systems that cause fatalities but far fewer than people do.
- Fairness. In data sets involving personally identifiable information, it is extremely important to ensure that there are no biases in terms of race, gender or ethnicity.
- Misuse. AI algorithms may be used for purposes other than those for which they were created. Wisniewski said these scenarios should be analyzed at the design stage to minimize the risks and introduce safety measures to reduce the adverse effects in such cases.

What are the benefits of ethical AI?

The rapid acceleration in AI adoption across businesses has coincided with -- and in many cases helped fuel -- two major trends: the rise of customer-centricity and the rise in social activism.

"Businesses are rewarded not only for providing personalized products and services but also for upholding customer values and doing good for the society in which they operate," said Sudhir Jha, senior vice president and head of the Brighterion unit at Mastercard.

AI plays a huge role in how consumers interact with and perceive a brand. Responsible use is necessary to ensure a positive impact. In addition to consumers, employees want to feel good about the businesses they work for. "Responsible AI can go a long way in retaining talent and ensuring smooth execution of a company's operations," Jha said.

Perspectives on an AI code of ethics

ARIJIT SENGUPTA
founder and CEO, Aible

"The fundamental problem with an AI code of ethics is that it's reactive, not proactive. We tend to define things like bias and go looking for bias and trying to eliminate it—as if that's possible."

SUDHIR JHA
senior vice president, head of the Brighterion unit, Mastercard

"Responsible AI can go a long way in retaining talent and ensuring smooth execution of a company's operations."

JASON SHEPHERD
vice president, ecosystem, Zededa

"Ultimately, consumers' willingness to proactively take control of their data and pay attention to potential threats enabled by AI is a complex equation based on a combination of instant gratification, value, perception and risk."

ADAM WISNIEWSKI
CTO and co-founder, AI Clearing

"Asimov's Laws of Robotics have to be practically applied in this new AI-powered reality. ... AI needs to have a strong degree of traceability to ensure that if harms arise, they can be traced back to the cause."

KELLY COMBS
director, Digital Lighthouse, KPMG

"[When developing an AI code of ethics] it's imperative to include clear guidelines on how the technology will be deployed and continuously monitored."

As AI and machine learning are becoming central to IT systems, companies must ensure their use of AI is ethical.

<u>What is an AI code of ethics?</u>

A proactive approach to ensuring ethical AI requires addressing three key areas, according to Jason Shepherd, vice president of ecosystem at Zededa, an edge AI tools provider.

- **Policy**. This includes developing the appropriate framework for driving standardization and establishing regulations. Efforts like the Asilomar AI Principles are essential to start the conversation, and there are several efforts spinning up around policy in Europe, the U.S. and elsewhere. Ethical AI policies also need to address how to deal with legal issues when something goes wrong. Companies may incorporate AI policies into their own code of conduct. But effectiveness will depend on employees following the rules, which may not always be realistic when money or prestige are on the line.
- **Education**. Executives, data scientists, front-line employees and consumers all need to understand policies, key considerations and potential negative impacts of unethical AI and fake data. One big concern is the tradeoff between ease of use around data sharing and AI automation and the potential negative repercussions of oversharing or adverse automations. "Ultimately, consumers' willingness to proactively take control of their data and pay attention to potential threats enabled by AI is a complex equation based on a combination of instant gratification, value, perception and risk," Shepherd said.
- **Technology**. Executives also need to architect AI systems to automatically detect fake data and unethical behavior. This requires not just looking at a company's own AI but

vetting suppliers and partners for the malicious use of AI. Examples include the deployment of deep fake videos and text to undermine a competitor, or the use of AI to launch sophisticated cyberattacks. This will become more of an issue as AI tools become commoditized. To combat this potential snowball effect, organizations need to invest in defensive measures rooted in open, transparent and trusted AI infrastructure. Shepherd believes this will give rise to the adoption of trust fabrics that provide a system-level approach to automating privacy assurance, ensuring data confidence and detecting unethical use of AI.

Examples of AI codes of ethics

An AI code of ethics can spell out the principles and provide the motivation that drives appropriate behavior. For example, Mastercard's Jha said he is currently working with the following tenets to help develop the company's current AI code of ethics:

- An ethical AI system must be inclusive, explainable, have a positive purpose and use data responsibly.
- An inclusive AI system is unbiased and works equally well across all spectra of society. This requires full knowledge of each data source used to train the AI models in order to ensure no inherent bias in the data set. It also requires a careful audit of the trained model to filter any problematic attributes learned in the process. And the models need to be closely monitored to ensure no corruption occurs in the future as well.

Ethics of AI: Safeguarding Humanity

Data privacy. Intrinsic bias. Robot rights. As artificial intelligence evolves, so do the many controversies that surround the use of this advanced technology. From military drones to shopping recommendations, AI is powering a wide array of smart products and services across nearly every industry—and with it, creating new ethical dilemmas for which there are no easy answers.

As technology continues to develop at an unprecedented rate, those involved with AI often lack the tools and knowledge to expertly navigate ethical challenges. In response, MIT Professional Education is pleased to introduce an exciting new course, **Ethics of AI: Safeguarding Humanity**.

Led by MIT experts Bernhardt L. Trout and Stefanie Jegelka, this course examines today's most pressing ethical issues related to AI and explores ways that organizations can leverage technology to benefit mankind. Over the course of two intensive days, you will:

- Deepen your understanding of the technological basis of AI
- Explore key ethical issues related to the technology's production and implementation

- Examine the relationship between AI and politics, from warfare to the manipulation of public opinion
- Analyze machine bias and other ethical risks
- Assess the individual and corporate responsibilities related to AI deployment

Ultimately, you will redirect your thinking from what is merely advantageous to what is genuinely good—and return to your workplace prepared to help your organization navigate the ethical aspects of AI development and deployment.

Deployment Pipeline:

In software development, a deployment pipeline is a system of automated processes designed to quickly and accurately move new code additions and updates from version control to production. In past development environments, manual steps were necessary when writing, building, and deploying code.

Main Stages of a Deployment Pipeline

There are four main stages of a deployment pipeline:

1. **Version Control**
2. **Acceptance Testsa**
3. **Independent Deployment**
4. **Production Deployment**

Version Control is the first stage of the pipeline. This occurs after a developer has completed writing a new code addition and committed it to a source control repository such as GitHub. Once the commit has been made, the deployment pipeline is triggered and the code is automatically compiled, unit tested, analyzed, and run through installer creation. If and when the new code passes this version control stage, binaries are created and stored in an artifact repository. The validated code then is ready for the next stage in the deployment pipeline.

In the **Acceptance Tests** stage of the deployment pipeline, the newly compiled code is put through a series of tests designed to verify the code against your team's predefined acceptance criteria. These tests will need to be custom-written based on your company goals and user expectations for the product. While these tests run automatically once integrated within the deployment pipeline, it's important to be sure to update and modify your tests as needed to consistently meet rising user and company expectations.

Once code is verified after acceptance testing, it reaches the **Independent Deployment** stage where it is automatically deployed to a development environment. The development environment should be identical (or as close as possible) to the production environment in order to ensure an accurate representation for functionality tests. Testing in a development environment allows teams to squash any remaining bugs without affecting the live experience for the user.

The final stage of the deployment pipeline is **Production Deployment**. This stage is similar to what occurs in Independent Deployment, however, this is where code is made live for the user rather than a separate development environment. Any bugs or issues should have been resolved at this point to avoid any negative impact on user experience. DevOps or operations typically handle this stage of the pipeline, with an ultimate goal of zero downtime. Using Blue/Green Drops or Canary Releases allows teams to quickly deploy new updates while allowing for quick version rollbacks in case an unexpected issue does occur.

Benefits of a Deployment Pipeline

Building a deployment pipeline into your software engineering system offers several advantages for your internal team, stakeholders, and the end user. Some of the primary benefits of an integrated deployment pipeline include:

- Teams are able to release new product updates and features much faster.
- There is less chance of human error by eliminating manual steps.
- Automating the compilation, testing, and deployment of code allows developers and other DevOps team members to focus more on continuously improving and innovating a product.
- Troubleshooting is much faster, and updates can be easily rolled back to a previous working version.
- Production teams can better respond to user wants and needs with faster, more frequent updates by focusing on smaller releases as opposed to large, waterfall updates of past production systems.

How to Build a Deployment Pipeline

A company's deployment pipeline must be unique to their company and user needs and expectations, and will vary based on their type of product or service. There is no one-size-fits-all approach to creating a deployment pipeline, as it requires a good amount of upfront planning and creation of tests.

When planning your deployment pipeline, there are three essential components to include:

- **Build Automation (Continuous Integration):** Build automation, also referred to as Continuous Integration or CI for short, are automated steps within development designed for continuous integration – the compilation, building, and merging of new code.

- **Test Automation:** Test automation relies on the creation of custom-written tests that are automatically triggered throughout a deployment pipeline and work to verify new compiled code against your organization's predetermined acceptance criteria.
- **Deploy Automation (Continuous Deployment/Delivery):** Like continuous integration, deploy automation with Continuous Deployment/Delivery (CD for short) helps expedite code delivery by automating the process of releasing code to a shared repository, and then automatically deploying the updates to a development or production environment.

When building your deployment pipeline, the primary goal should be to eliminate the need for any manual steps or intervention. This means writing custom algorithms for automatically compiling/building, testing, and deploying new code from development. By taking these otherwise tedious and repetitive steps off developers and other DevOps team members, they can focus more on creating new, innovative product updates and features for today's highly competitive user base.

Model Serving:

The basic meaning of model serving is to host machine-learning models (on the cloud or on premises) and to make their functions available via API so that applications can incorporate AI into their systems. Model serving is crucial, as a business cannot offer AI products to a large user base without making its product accessible. Deploying a machine-learning model in production also involves resource management and model monitoring including operations stats as well as model drifts.

Model Performance Monitoring:

Model performance monitoring is a **basic operational task that is implemented after an AI model has been deployed**. ML teams need a strategy to quickly adapt ML models to the constantly changing patterns in real-world data. What Is Model Monitoring? Machine learning model monitoring is the tracking of an ML model's performance in production.

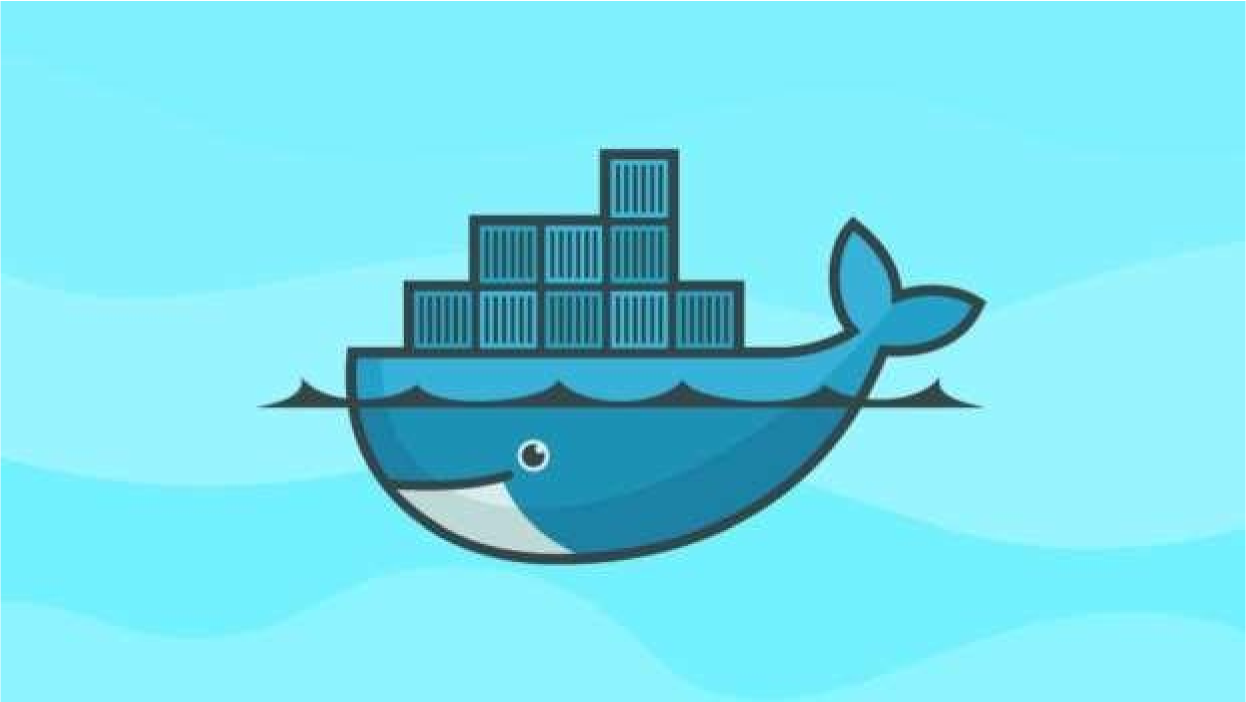
Deploying ML models as Docker Containers:

Introduction on Docker

Docker is everywhere in the world of the software industry today. Docker is a DevOps tool and is very popular in the DevOps and MLOPS world. Docker has stolen the hearts of many developers, system administrators, and engineers, among others.

We'll start with what Docker brings you; why you should use it in the first place. The article is chopped into parts this is because I'm trying to explain as simply as I can in order to make it accessible for as many people as possible. So now let's go!

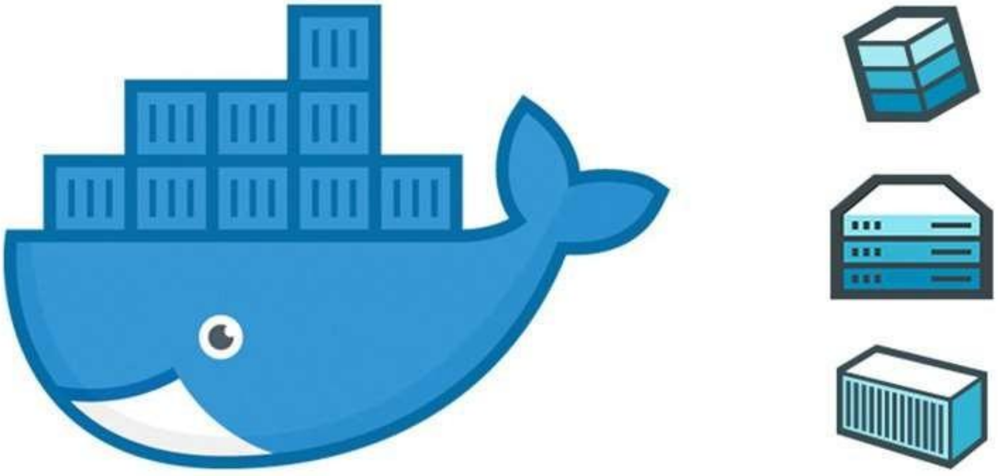
What is Docker?



When working on a team project we often have to see how each other code is running in our system, well I am sure you have ever said this statement or listened to this statement “This code is not running in my machine” or “I don’t know this is running in my computer but not running in your’s computer(another user)”. So these types of things can be fixed easily by docker.

Docker is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime. Using Docker, you can quickly deploy and scale applications into any environment and know your code will run.

What are Containers?



Docker provides the ability to package and run an application in a loosely isolated environment called a container. The isolation and security allow you to run many containers simultaneously on a given host. Containers are lightweight and contain everything needed to run the application, so you do not need to rely on what is currently installed on the host. You can easily share containers while you work, and be sure that everyone you share with gets the same container that works in the same way.

Why Use Containers for Machine Learning?

1. Running an ML model on the computer is an easy task. But when we want to use that model at the production stage in other systems, it’s a complex task. Docker makes this task easier, faster, and more reliable.

2. Using Docker we can easily reproduce the working environment to train and run the model on different operating systems

3. We can easily deploy and make your model available to the clients using technologies such as OpenShift, a Kubernetes distribution.

4. Developers can keep track of different versions of a container image, check who built a version with what, and also roll back to previous versions.

5. Even if our Machine Learning application is down, repairing, or updating, it will not stop running.

6. Our machine learning model is usually written in a single programming language such as python but the application will certainly need to interact with other applications written in different programming languages. Docker manages all these interactions as each microservice

can be written in a different language allowing scalability and the easy addition or deletion of independent services.

How to Deploy the ML Model Inside a Docker Container?

Let us understand how to deploy our Machine Learning model inside a Docker container. Here, I will take a simple Titanic dataset Machine Learning model to illustrate the workflow.

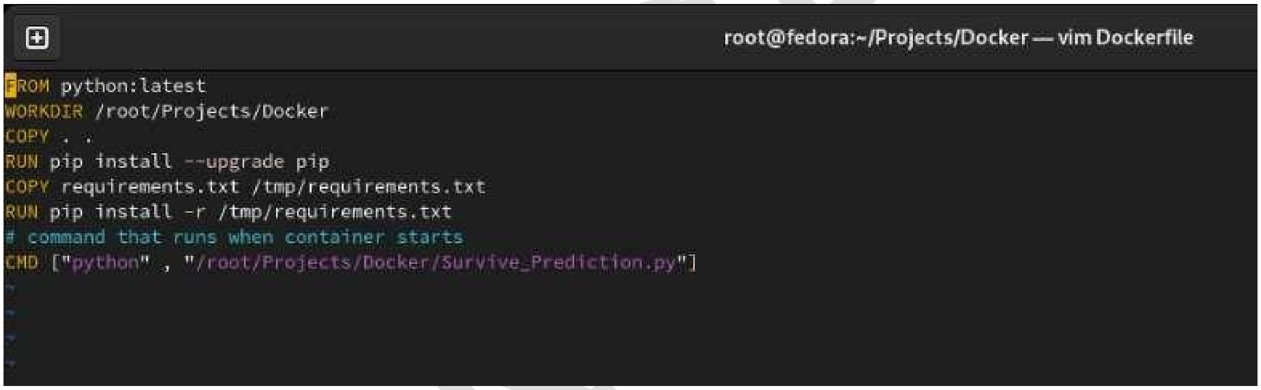
1. Create a separate directory for this task and copy your Machine learning code to that directory.

2. Create a **Dockerfile.**

What’s a Dockerfile?

It’s just a way to create your own customized Docker image. This file contains step-by-step requirements as per our use case. Simply Dockerfile is a script on a recipe for creating a Docker image. It contains some. special keywords such as FROM, RUN, CMD, etc.

Dockerfile is dynamic in nature. It means at any point in time if you want to change any stop, update or add anything, you can just add & build. That’s quick & time-saving.



```
FROM python:latest
WORKDIR /root/Projects/Docker
COPY . .
RUN pip install --upgrade pip
COPY requirements.txt /tmp/requirements.txt
RUN pip install -r /tmp/requirements.txt
# command that runs when container starts
CMD ["python" , "/root/Projects/Docker/Survive_Prediction.py"]
~
~
~
~
```

Now let us understand the code inside Dockerfile

FROM This is used for providing the name of the base image on which we’ll be adding our requirements. Here, I have used Python as a base image for the container.

COPY command will copy the specified files from the local machine to the container that will be launched using this image.

RUN it’s a build time keyword, & any program that goes with it will be executed during the building of the image.

CMD It’s a runtime keyword. Any program one command goes with it will be executed when the container is launched.

NOTE: In **Docker or Container** world, we launch a specific container for a specific program or process only. So after it is Executed completely – we don’t need the environment Hence we can conclude the life of the process=Life of the container

Entrypoint and CMD, both can be used to specify the command to be executed when the container is started. The only difference is that **ENTRYPOINT** doesn’t allow you to override the command. Instead, anything added to the end of the docker **RUN** command is appended to the command.

3. Python Code

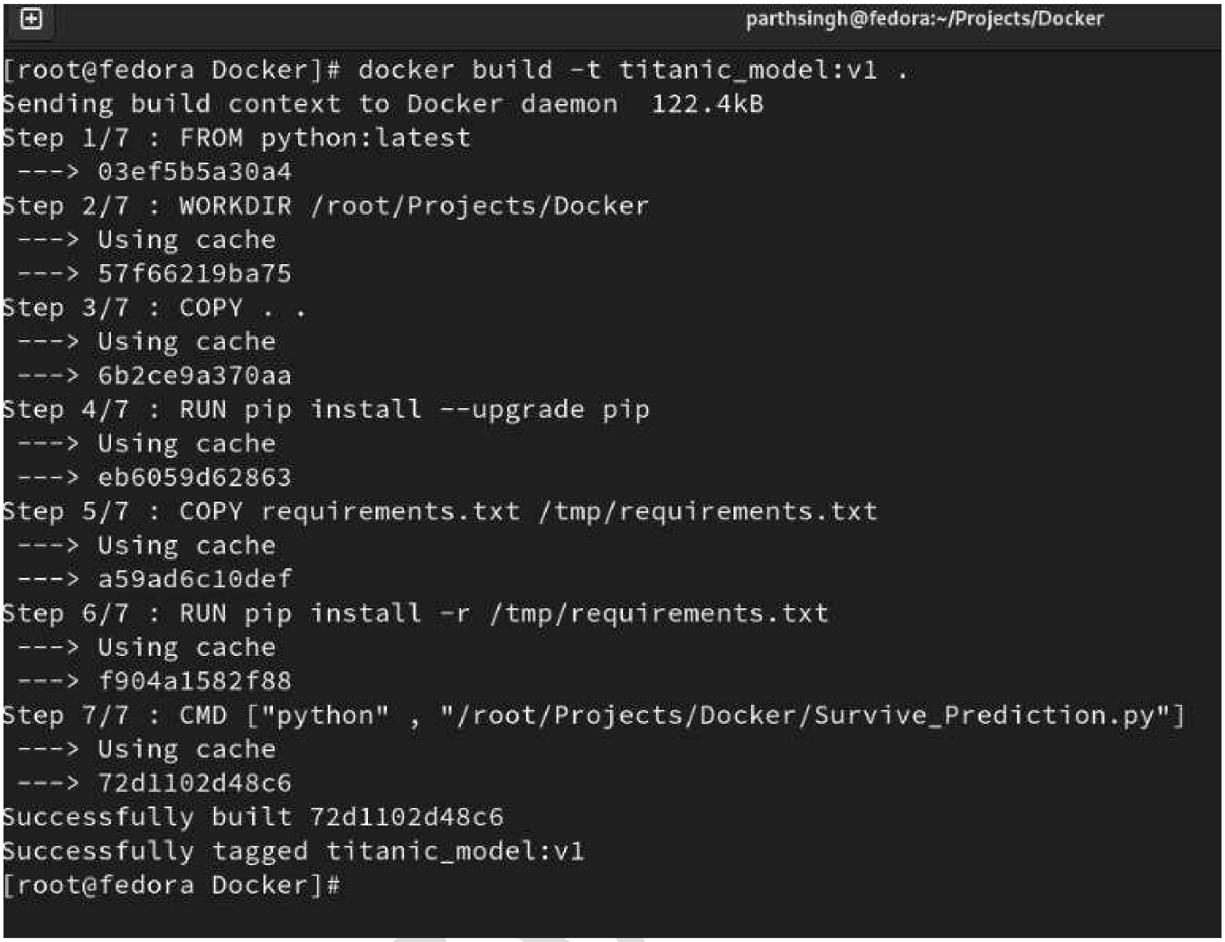
This python code will be run when as soon as our container start. Here I have used the joblib module in python through which we can save and load our trained models.

```
import joblib



classifier = joblib.load('survive_prediction.pkl')

print("Enter the following details to make the predictions:- n")



pclass = int(intput("Enter The Pclass:- "))

Age = int(intput("Enter The Age:- "))

SibSP = int(intput("Enter The SibSp:- "))

Parch = int(intput("Enter The Parch:- "))

Sex = int(intput("Enter The Sex:- "))


passenger_prediction = classifier.predict([[pclass,Age,SibSP,Parch,Sex]])


if passenger_prediction == 0:

print("Not Survived.")

else ;


print("Survived")
```

4. Now, we will going to build the image from the Dockerfile that we have created just above.

To build the image we use the following command.

```
docker build -t image_name:version .
```



```
[root@fedora Docker]# docker build -t titanic_model:v1 .
Sending build context to Docker daemon  122.4kB
Step 1/7 : FROM python:latest
 ---> 03ef5b5a30a4
Step 2/7 : WORKDIR /root/Projects/Docker
 ---> Using cache
 ---> 57f66219ba75
Step 3/7 : COPY . .
 ---> Using cache
 ---> 6b2ce9a370aa
Step 4/7 : RUN pip install --upgrade pip
 ---> Using cache
 ---> eb6059d62863
Step 5/7 : COPY requirements.txt /tmp/requirements.txt
 ---> Using cache
 ---> a59ad6c10def
Step 6/7 : RUN pip install -r /tmp/requirements.txt
 ---> Using cache
 ---> f904a1582f88
Step 7/7 : CMD ["python" , "/root/Projects/Docker/Survive_Prediction.py"]
 ---> Using cache
 ---> 72d1102d48c6
Successfully built 72d1102d48c6
Successfully tagged titanic_model:v1
[root@fedora Docker]#
```

5. Now Finally, we are ready to launch our container and run our machine learning model

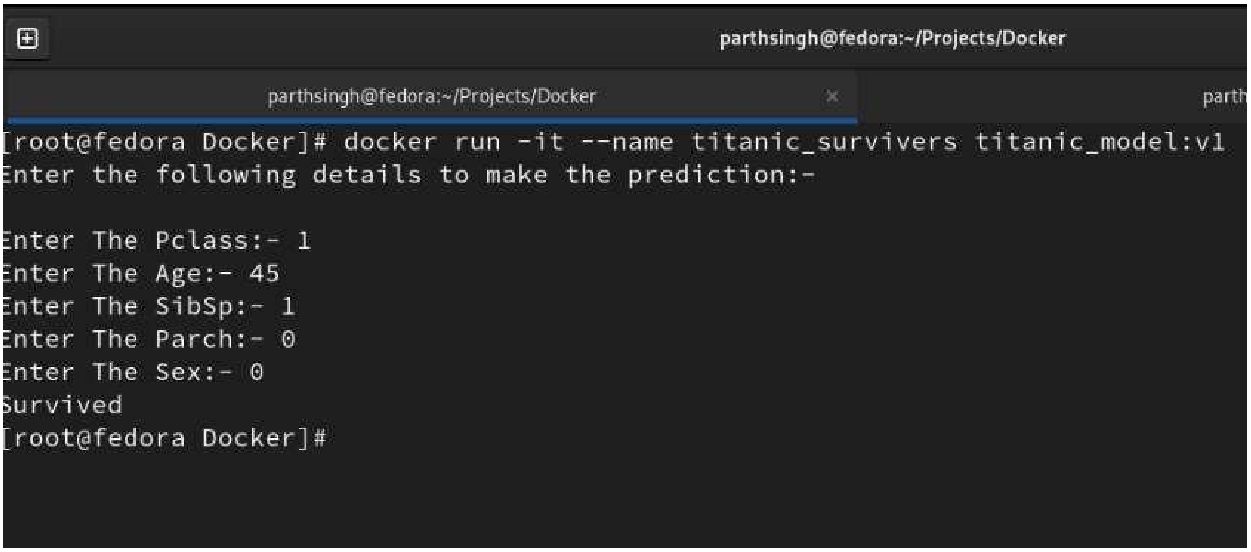
docker run -it –name titanic_survivers titanic_model:v1

-> When we run this command, a new environment is launched; a new OS entirely. So behind the scene, docker-engine does a lot of tasks such as providing network card, storage, complete new file system, RAM/CPU, etc. These are with respect to an OS.

->So if you want to see the entire details of the container you can use

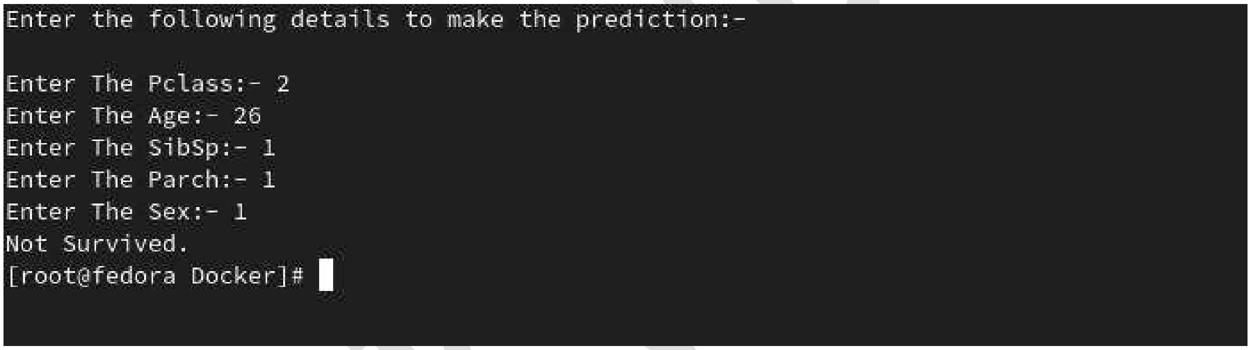
docker info container_name

-> With this command, you can see the entire details of the container like, how much storage it is using, what’s the network card and many more.


parthsingh@fedora:~/Projects/Docker
parthsingh@fedora:~/Projects/Docker
parth

```
[root@fedora Docker]# docker run -it --name titanic_survivers titanic_model:v1
Enter the following details to make the prediction:-

Enter The Pclass:- 1
Enter The Age:- 45
Enter The SibSp:- 1
Enter The Parch:- 0
Enter The Sex:- 0
Survived
[root@fedora Docker]#
```

6. Using CLI(Command Line Input) we can give input to our python code. By command-line input means through the keyboard we can pass input to the container.



```
Enter the following details to make the prediction:-

Enter The Pclass:- 2
Enter The Age:- 26
Enter The SibSp:- 1
Enter The Parch:- 1
Enter The Sex:- 1
Not Survived.
[root@fedora Docker]# 
```

Conclusion

I hope now you have some understanding of how to deploy your model inside the docker/ containerize model in docker. Still, we’ve only scratched the surface when it comes to all of the benefits Docker has to offer