

Natural Processing Language

Riya Bihani

Natural Language Processing

Natural Language Processing, or NLP for short, is a field of Artificial Intelligence (AI) that makes human language intelligible to machines. NLP combines the power of linguistics and computer science to study the rules and structure of language, and create intelligent systems (run on machine learning and NLP algorithms) capable of understanding, analyzing, and extracting meaning from text and speech. It's at the core of tools we use every day – from translation software, chatbots, spam filters, and search engines, to grammar correction software, voice assistants, and social media monitoring tools.

Importance of NLP

Businesses use massive quantities of unstructured, text-heavy data and need a way to efficiently process it. A lot of the information created online and stored in databases is natural human language, and until recently, businesses could not effectively analyze this data. This is where natural language processing is useful.

Uses of NLP

- Search, spell checking, keyword search, finding synonyms, complex questions answering
- Extracting information from websites such as: products, price, dates, locations, people or names
- Machine translation (i.e. Google translate), speech recognition, personal assistants (think about Amazon Alexa, Apple Siri, Facebook M, Google Assistant or Microsoft Cortana)
- Chat bots/dialog agents for customer support, controlling devices, ordering goods
- Matching online advertisements, sentiment analysis for marketing or finance/trading
- Identifying financial risks or fraud

Sentimental Analysis

Sentiment Analysis (also known as opinion mining or emotion AI) is a sub-field of NLP that tries to identify and extract opinions within a given text across blogs, reviews, social media, forums, news etc. Sentiment Analysis can help craft all this exponentially growing unstructured text into structured data using NLP and open source tools.

For example –

Suppose, there is a fast-food chain company and they sell a variety of different food items like burgers, pizza, sandwiches, milkshakes, etc. They have created a website to sell their food and now the customers can order any food item from their website and they can provide reviews as well, like whether they liked the food or hated it.

User Review 1: I love this cheese sandwich, it's so delicious.

User Review 2: This chicken burger has a very bad taste.

So, as we can see that out of these above 2 reviews,

The first review is definitely a positive one and it signifies that the customer was really happy with the sandwich.

The second review is negative, and hence the company needs to look into their burger department.

In this problem statement, we will be using Sentimental Analysis to classify the reviews into positive and negative.

Preprocessing

To preprocess your text simply means to bring your text into a form that is predictable and analyzable for your task.

1. Remove Punctuations, Numbers

Punctuations, Numbers doesn't help much in processing the given text, if included, they will just increase the size of bag of words that we will create as last step and decrease the efficiency of algorithm.

2. Convert to Lowercase

Converting character to the same case so the same words are recognized as the same. In this case we converted to lowercase. For example 'Good' becomes 'good'

3. Tokenization

It involves splitting sentences and words from the body of the text. It is the process of converting text into tokens before transforming it into vectors. It is easier to filter out unnecessary tokens.

4. Stopwords

Now, we remove the stopwords. Stop words are a set of commonly used words in a language. Examples of stop words in English are "a", "the", "is", "are" and etc. The intuition behind using stop words is that, by removing low information words from text, we can focus on the important words instead. You can even customize lists of stopwords to include words that you want to ignore.

5. Stemming/Lemmatization

- Both of these words mean to take roots of the word
- Stemming usually refers to a process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes.
- Lemmatization usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma.
- For example – The word "better" is transformed into the word "good" by a lemmatizer but is unchanged by stemming.
- Even though stemmers can lead to less-accurate results, they are easier to build and perform faster than lemmatizers. But lemmatizers are recommended if you're seeking more precise linguistic rules.

6. Check for null and empty values

7. Making the bag of words

- Bag of words is a Natural Language Processing technique of text modelling.
- A bag of words is a representation of text that describes the occurrence of words within a document. We just keep track of word counts and disregard the grammatical details and the word order.
- One of the biggest problems with text is that it is messy and unstructured, and machine learning algorithms prefer structured, well defined fixed-length inputs and by using the Bag-of-Words technique we can convert variable-length texts into a fixed-length vector.
- We can use the `CountVectorizer()` function from the Sk-learn library to easily implement the BoW model using Python.
- `CountVectorizer` is a great tool provided by the scikit-learn library in Python. It is used to transform a given text into a vector on the basis of the frequency (count) of each word that occurs in the entire text. This is helpful when we have multiple such texts, and we wish to convert each word in each text into vectors (for using in further text analysis).
- `CountVectorizer` creates a matrix in which each unique word is represented by a column of the matrix, and each text sample from the document is a row in the matrix. The value of each cell is nothing but the count of the word in that particular text sample.

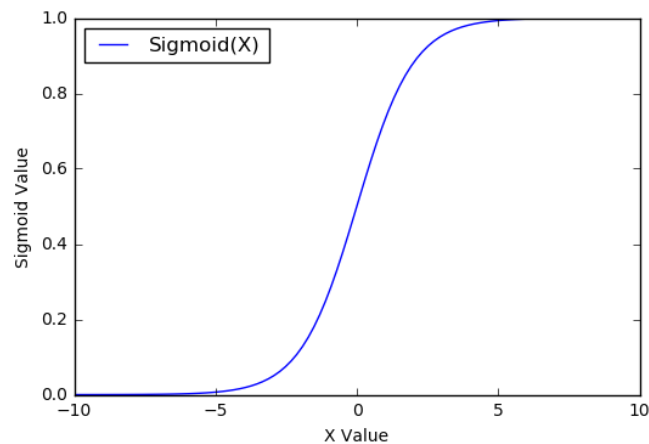
Techniques / Algorithms

1. Logistic Regression

Logistic Regression is one of the few algorithms that is used for the task of Classification of data.

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

For the sake of simplicity, assume we have only one data point. It has a total of n features. Also assume that we have the appropriate weight matrix. Now given a data point we need to predict a class label (Classify as 1 or 0). This output is fed to a Sigmoid function to get a value between 0 and 1. Let's call this probability that the predicted class is 1. If the probability is greater than .5 then the predicted class is 1. If the probability is less than .5 then that predicted class is 0.



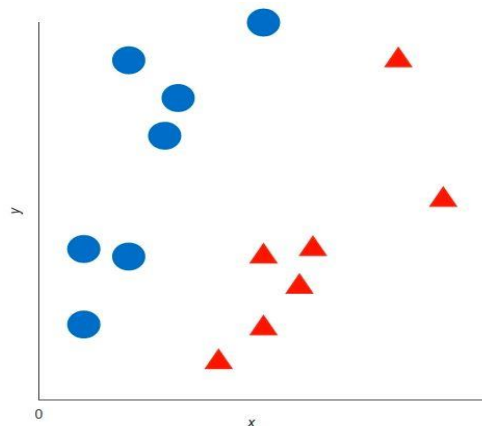
So if the input belongs to a positive class, then the dot product should be a larger value which in turn is sent to the sigmoid to generate a greater probability for positive class and vice versa.

2. Support Vector Machines (SVM)

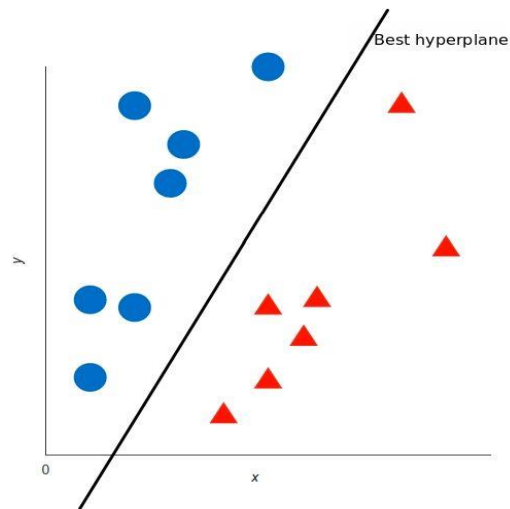
Support vector machines is an algorithm that determines the best decision boundary between vectors that belong to a given group (or category) and vectors that do not belong to it. It can be applied to any kind of vectors which encode any kind of data.

So, when SVM determines the decision boundary we mentioned above, SVM decides where to draw the best “line” (or the best hyperplane) that divides the space into two subspaces: one for the vectors which belong to the given category and one for the vectors which do not belong to it.

So, provided we can find vector representations which encode as much information from our texts as possible, we will be able to apply the SVM algorithm to text classification problems and obtain very good results.



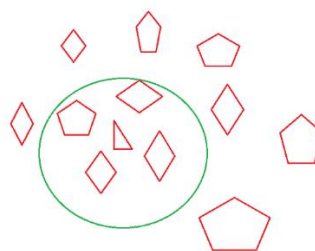
Say, for example, the blue circles in the graph below are representations of training texts which talk about the Pricing of a SaaS Product and the red triangles are representations of training texts which do not talk about that.



Now that the algorithm has determined the decision boundary for the category you want to analyze, you only have to obtain the representations of all of the texts you would like to classify and check what side of the boundary those representations fall into.

3. K Nearest Neighbor

K Nearest Neighbor is one of the fundamental algorithms in machine learning. Machine learning models use a set of input values to predict output values. KNN is one of the simplest forms of machine learning algorithms mostly used for classification. It classifies the data point on how its neighbor is classified.



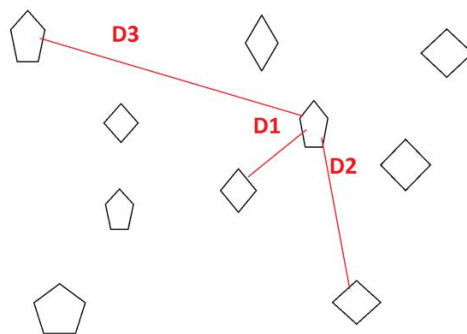
k = 4

KNN classifies the new data points based on the similarity measure of the earlier stored data points. For example, if we have a dataset of tomatoes and bananas. KNN will store similar measures like shape and color. When a new

object comes it will check its similarity with the color (red or yellow) and shape.

K in KNN represents the number of the nearest neighbors we used to classify new data points. Choosing the right value of K is called parameter tuning and it's necessary for better results

- a. $K = \text{sqrt}(\text{total number of data points})$.
- b. Odd value of K is always selected if the number of classes is two.



We usually use Euclidean distance to calculate the nearest neighbor. If we have two points (x, y) and (a, b). The formula for Euclidean distance (d) will be

$$d = \text{sqrt}((x-a)^2 + (y-b)^2)$$

4. Naïve Bayes Classifier

Naive Bayes classifier is a straightforward and powerful algorithm for the classification task. Even if we are working on a data set with millions of records with some attributes, it is suggested to try Naive Bayes approach.

Naive Bayes is a kind of classifier which uses the Bayes Theorem. This theorem works on conditional probability. Conditional probability is the probability that something will happen, given that something else has already occurred. Using the conditional probability, we can calculate the probability of an event using its prior knowledge.

Below is the formula for calculating the conditional probability.

$$P(A | B) = \frac{P(B | A) \cdot P(A)}{P(B)}$$

A, B = events

$P(A|B)$ = probability of A given B is true

$P(B|A)$ = probability of B given A is true

$P(A), P(B)$ = the independent probabilities of A and B

There are a few types of Naïve Bayes Classifier such as –

i. Gaussian Naïve Bayes

A Gaussian Naive Bayes algorithm is a special type of NB algorithm. It's specifically used when the features have continuous values. It's also assumed that all the features are following a gaussian distribution i.e, normal distribution.

ii. MultiNomial Naive Bayes

MultiNomial Naive Bayes is preferred to use on data that is multinomially distributed. It is one of the standard classic algorithms which is used in text categorization (classification). Each event in text classification represents the occurrence of a word in a document.

Confusion Matrix

A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. This gives us a holistic view of how well our classification model is performing and what kinds of errors it is making.

For a binary classification problem, we would have a 2 x 2 matrix.

		ACTUAL VALUES	
		POSITIVE	NEGATIVE
PREDICTED VALUES	POSITIVE	TP	FP
	NEGATIVE	FN	TN

True Positive (TP)

- The predicted value matches the actual value
- The actual value was positive and the model predicted a positive value

True Negative (TN)

- The predicted value matches the actual value
- The actual value was negative and the model predicted a negative value







False Positive (FP) – Type 1 error

- The predicted value was falsely predicted
- The actual value was negative but the model predicted a positive value
- Also known as the Type 1 error

False Negative (FN) – Type 2 error

- The predicted value was falsely predicted
- The actual value was positive but the model predicted a negative value
- Also known as the Type 2 error

For a 3 x 3 matrix,

		ACTUAL VALUES		
				
PREDICTED VALUES		+ve 1	-ve 2	-ve 3
		-ve 4	+ve 5	-ve 6
		-ve 7	-ve 8	+ve 9

The true positive, true negative, false positive and false negative for each class would be calculated by adding the cell values as follows:

Facebook

$$TP = Cell_1$$

$$FP = Cell_2 + Cell_3$$

$$TN = Cell_5 + Cell_6 + Cell_7 + Cell_8$$

$$FN = Cell_4 + Cell_9$$

Instagram

$$TP = Cell_5$$

$$FP = Cell_4 + Cell_6$$

$$TN = Cell_1 + Cell_3 + Cell_7 + Cell_9$$

$$FN = Cell_2 + Cell_8$$

Snapchat

$$TP = Cell_9$$

$$FP = Cell_7 + Cell_8$$

$$TN = Cell_1 + Cell_2 + Cell_4 + Cell_5$$

$$FN = Cell_3 + Cell_6$$

Similarly, TP, TN, FP and FN can be calculated for any N x N matrix.

Precision, Recall and F1-Score

- Precision tells us how many of the correctly predicted cases actually turned out to be positive. This would determine whether our model is reliable or not. It is a useful metric in cases where False Positive is a higher concern than False Negatives.

$$Precision = \frac{TP}{TP + FP}$$

- Recall tells us how many of the actual positive cases we were able to predict correctly with our model. It is a useful metric in cases where False Negative trumps False Positive.

$$Recall = \frac{TP}{TP + FN}$$

- In practice, when we try to increase the precision of our model, the recall goes down, and vice-versa. The F1-score captures both the trends in a single value.

$$F1 - score = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}}$$

F1-score is a harmonic mean of Precision and Recall, and so it gives a combined idea about these two metrics. But the interpretability of the F1-score is poor. This means that we do not know what our classifier is maximizing – precision or recall? So, we use it in combination with other evaluation metrics which gives us a complete picture of the result.

Conclusion

According to me, the algorithm to be applied to solve this problem will be the Multinomial Naïve Bayes as it is one of the most popular supervised learning classifications that is used for the analysis of the categorical text data. It is highly scalable and can easily handle large datasets. It is also easy to implement as you only have to calculate probability. It also gives the most accurate response out of all the other solutions.