Know What’s Real

Identification of Disaster-related tweets using Natural Language Processing

Santosh Jonnakuti, Danam Yashwanth Kumar, L.S Yashwanth, Chokkari Dinesh, Bathini Hemanth

Abstract

In today’s world social media has become an integral part of life. Twitter is a social networking portal which provides the users a platform to post news, data and thoughts. Twitter has become an essential mode of communication medium during the occurrence of an emergency or disaster. This information regarding disasters propagated over social media can save thousands of lives by alerting others so that they can take evasive action. Many agencies are trying to programmatically analyze tweets and recognize disasters and emergencies. This kind of work can be beneficial to millions of people connected to the internet, who can be alerted in the case of an emergency or disaster. But the real challenge lies in segregating the tweets trying to announce a disaster from the ones which are not related to a disaster.Twitter data is unstructured data, thus Natural Language Processing (NLP) has to be performed on the twitter data to classify them into classes –“Disaster” and “Not Disaster ''. The paper does a prediction on the test set. This paper uses four machine learning algorithms for building the classifier model and for making predictions.

Objective

As already stated before, this paper’s main objective is to analyze a twitter dataset and subsequently classify the tweets as “Disaster” or “Not Disaster”. The dataset is cleaned and tokenized(Data Preprocessing). It is divided into training set and testing set for accuracy checking purposes. This work is in the field of Natural Language Processing(NLP).

Source of the Data

The data used for this in this paper has been retrieved from the kaggle portal. It is a competition dataset related to NLP work. The dataset has two files train.csv to train and test.csv to test the classifier. The train dataset has 7613 tuples (rows) which represent 7613 tweets.

The Data given in the Kaggle Dataset has following Columns :
id - a unique identifier for each tweet
text - the text of the tweet
location - the location the tweet was sent from (may be blank)
keyword - a particular keyword from the tweet (may be blank)
target - in train.csv only, this denotes whether a tweet is about a real disaster (1) or not (0).

Data Pre-Processing

The tweets from the dataset have to undergo data pre-processing and data cleaning. Data pre-processing and cleaning involves the mechanisms of corpus creation, lower-case conversion, stop-words removal, punctuation removal, URL removal, username removal, leading and ending blank spaces removal and text stemming.

A corpus is basically a collection of text data. The tweets are converted into a corpus for further processing and analysis. The dataset of tweets contain a large number of fields – id, text, username, location. From this large dataset only the text is needed for analysis and prediction. So the dataset is converted into a text corpus holding the tweet text pieces.

After the creation of corpus some amount of data cleaning is required. The tweets have a mixture of upper-case and lower-case letters. The entire text corpus needs to be converted into lower-case for the sake of uniformity. Then the stop-words present in the dataset have to be removed. Stop-words are basically conjunctions and prepositions which have no effect on the overall impact or opinion of the text. The punctuations also need to be removed from the corpus for exactly the same reason.

The results of the above processes are used to perform stemming. Stemming is an important concept in text analysis. Stemming is a procedure of reducing a derived or inflected word to its basic form.

The classification algorithm that has been used in this paper are four machine learning algorithms, namely,

1. Naive Bayes Algorithm
2. Logistic Regression Algorithm
3. K Nearest Neighbours Algorithm
4. Decision Tree Algorithm

The accuracy of the trained model can be computed using the Accuracy score and by generating the Confusion Matrix.

Accuracy = (TP+TN) / (TP+FP+TN+FN)

The Confusion Matrix is used as a measure of a classifier model. It evaluates the performance of a model on test data for which the real reference values are already given. In this paper, the predicted class values are kept as rows and the actual class values (reference values) are kept in the columns. Using the values – TP (True Positive) , FP (False Positive) , TN (True Negative) and FN (False Negative) the Confusion matrix is generated .

Naive Bayes Algorithm

1. Naive Bayes algorithm is a supervised learning algorithm, which is based on **Bayes theorem** and used for solving classification problems.

2. **It is a probabilistic classifier, which means it predicts on the basis of the probability of an object**.

3. **Bayes Theorem :** P(A | B) = P(A)P(B | A)/P(B).

4. The Procedure to implement Naive Bayes as follows:
   - a. Convert the given dataset into frequency tables.
   - b. Generate Likelihood table by finding the probabilities of given features.
   - c. Now, use Bayes theorem to calculate the posterior probability.

5. The Naives Bayes Model we used to solve this is the **Guassian** model.

6. Python implementation of **Naive Bayes** model:
   - a. Data pre processing step.
   - b. Fitting Naive Bayes to the Training set.
   - c. Predicting the test result.
   - d. Test accuracy of the result(Creation of Confusion matrix).
   - e. Visualizing the test set result.

7. This Model yielded an accuracy of **78%**.

Logistic Regression

1. Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
2. Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, True or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1**.
3. Python implementation of **Logistic Regression** model:

a. Data pre processing step.
b. Fitting Logistic Regression to the Training set.
c. Predicting the test result.
d. Test accuracy of the result(Creation of Confusion matrix).
e. Visualizing the test set result.

4. Equation for **Logistic Regression**:

$$log\left[\frac{y}{1-y}\right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx$$

5. This Method yielded an accuracy of **81%**.

K Nearest Neighbours

1. K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
2. The K-NN algorithm assumes the similarity between the new case/data and available cases and puts the new case into the category that is most similar to the available categories.
3. K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.
4. Setting Value for **K**:
   a.There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them. The most preferred value for K is 5.
   b.A very low value for K such as K=1 or K=2, can be noisy and lead to the effects of outliers in the model.
5. Python implementation of **KNN:**
   a.Data Pre-processing step.
   b.Fitting the K-NN algorithm to the Training set.
   c.Predicting the test result.
   d.Test accuracy of the result(Creation of Confusion matrix).
   e.Visualizing the test set result.
6. This model yielded an accuracy of **74%**.

Decision Tree

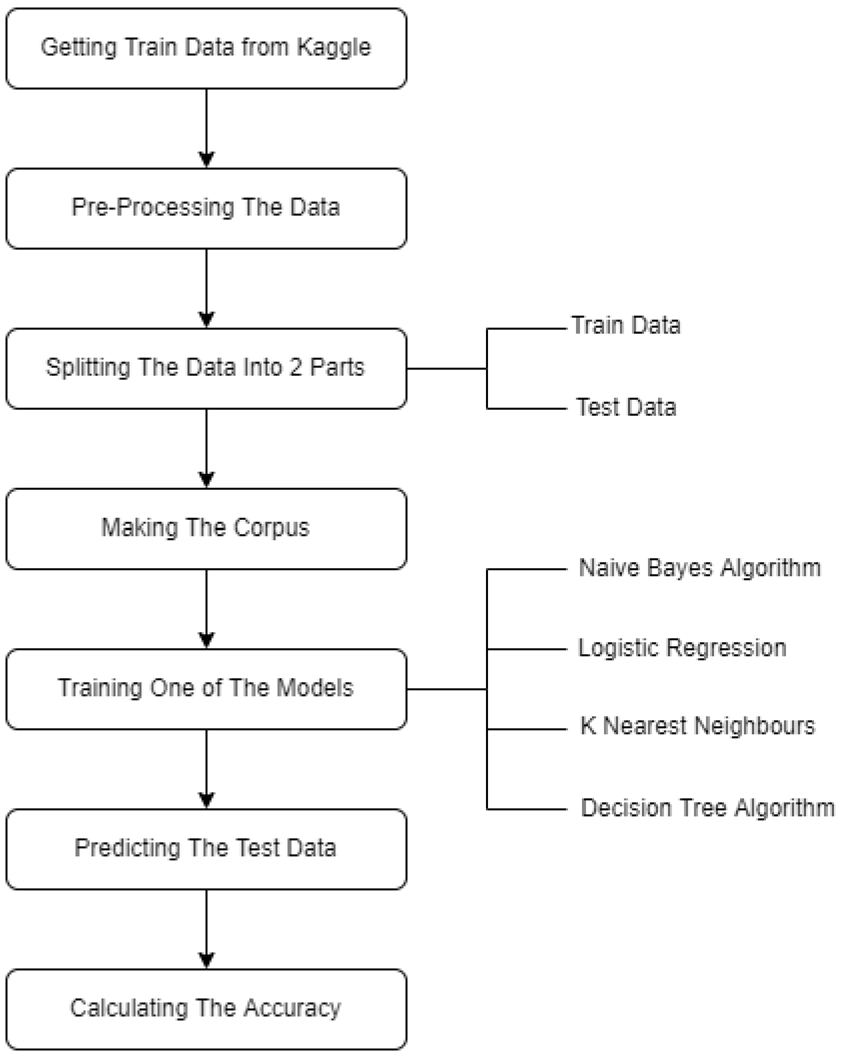
1. Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**
2. In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
3. ***It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.***
4. In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.**
5. **Attribute Selection Measures:**

   a.Information Gain: **Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each feature).**

   b.Gini index: **Gini Index=** $1-\sum_j P_j^2$

6. This method yielded an accuracy of **76%**.

Methodology of the experiment



Results and Conclusion