Sentiment Analysis : Predicting sentiment of COVID-19 tweets

Sonica Sinha, Mohd Taufique
Data science trainees,
AlmaBetter, Bangalore

Abstract:

Twitter is a social media platform where people used to share their information. On December 31, 2019 the first coronavirus started from China encountered by people all around the world. People all around the world in a state of confusion and fear started sharing their concerns on twitter via tweet. And the present dataset just contains the tweets of around one month between months of March & April, 2020 with five categories of sentiments.

Our experiment can help in detecting the different categories of sentiments being tweeted on Twitter. And this has been performed by development of Machine Learning algorithms using Supervised Classification.

Keywords:machine learning, sentiments, supervised classification, Twitter

1. Introduction

The world has suffered as well as encountered a major issue of Covid19 in recent years from the end of year 2019. In the state of chaos when people are staying in their homes all around the world, then the only source of sharing their concerns are on social media platforms. So, among these Twitter is a social media site where people show their concerns from different parts of the world. Some tweets showed anger, sadness, concern, and positivity towards their respective government agencies and family members living in different parts of the world. The tweets were tweeted on different dates with different moods.

Our goal here is to build a predictive model which can detect the sentiment associated with those tweets.

2.Problem Statement

The challenge of the problem is to build a supervised classification model which can predict the sentiment of COVID-19 tweets provided in our dataset. The tweets have been pulled from Twitter and manual tagging has been done then. This is a supervised ML classification problem.

Following columns has been provided with the dataset:

- **UserName** - identification given by twitter to the user.
- **ScreenName** - name projected on the screen of the user.
- **Location** - name of the location from which the tweet was shared.
- **TweetAt** - time of the tweet at which the tweet was shared.
- **OriginalTweet** - the original tweet shared by the user.

- **Sentiment** - defined sentiment or label from the shared tweet.

The **Sentiment** column is the dependent variable which consists of 5 different labels which are positive, negative, neutral, extremely positive, and extremely negative.

The **OriginalTweet** column is the independent variable and needs some preprocessing to fit our classification model.

3. Steps involved:

- **Data Cleaning & Null Value Treatment**

  After loading the dataset, head, tail, data type, shape were explored. Further , the dataset is checked for its null value and only the **Location** column contains the null value of 20.87%. As the treatment of location value was not that important to us so as for treatment purpose we did not include it in our next process except doing the exploration part. Hence we dropped them at the beginning of our project inorder to get a better result.

- **Exploratory Data Analysis**

  EDA was performed on 4 different columns like TweetAt, OriginalTweet, Location and Sentiment.
  - Location: Top 15 locations were explored in which London was the place from where maximum tweets had been done.
  - TweetAt: All together 30 different unique dates came up from mid March to mid April.
  - Original Tweet: Tweets containing different hashtags, punctuation marks, handle (@user), etc from all around the world.
  - Sentiment: Total 5 different sentiments were present i.e.Extremely positive, positive, neutral, negative and extremely negative.

  And henceforth, different kinds of plots were plotted like count plot, bar plot, heatmap, etc.

- **Text Preprocessing**

  Here from OriginalTweet feature we firstly removed unnecessary portions in following order, the procedure normally followed in NLP:
  - Removing handle (@user),
  - Punctuation marks, url, http,
  - Short words (english),
  - Stopwords,
  - Stemming
  - Tokenization

- **Word Cloud**

  Different most frequently occurring words in different Sentiments were plotted using WordCloud library, matplotlib and numpy.

  Same word cloud is prepared for the overall most frequent occurring words within the dataset of OriginalTweet column.

- **Feature Engineering**

  In feature engineering we firstly copied the data frame with just two columns from the main data frame having cleaned OriginalTweet and Sentiment. And further proceed in the following manner.

  - Encoded the Sentiments into 3 categories from 5, by merging the extremely positive & positive as Positive and extremely negative & negative as Negative and Neutral, respectively.
  - The value provided as Negative '-1', Neutral '0' and Positive '1'. Again the count value of positive sentiment is highest. As Sentiment is our target variable.

4. Vectorization:

Here we have used CountVectorizer() for converting the words into vectors.

Scikit-learn's CountVectorizer is used to convert a collection of text documents to a vector of term/token counts. It also enables the pre-processing of text data prior to generating the vector representation. This functionality makes it a highly flexible feature representation module for text.

5. Train-Test Split:

In the next step after vectorization, we proceeded for the train-test split by taking the ratio of 80:20 with random state value zero.

5. Model Training:

After the splitting the variables are fitted using different supervised classification machine learning models.

The used model for training are:

1. Logistic Regression
2. Stochastic Gradient Descent - Classifier (SGD-Classifier)
3. Random Forest Classifier
4. Support Vector Machine
5. Naive Bayes Classifier
6. CatBoost

- **Tuning the hyperparameters for better accuracy**

  Tuning the hyperparameters of respective algorithms is necessary for getting better accuracy and to avoid overfitting in case of tree based models like Random Forest Classifier, Logistic Regression and SGD-Classifier. Here we have used GridSearchCV for tuning our models.

6. Algorithms Used:

1. Logistic Regression:

Logistic regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.).

In other words, the logistic regression model predicts
P(Y=1) as a function of X.
The function used in Logistic Regression is sigmoid function or the logistic function given by:

f(x)= 1/1+e ^(-x)

Types of Logistic Regression:

- Binary Logistic Regression: The target variable has only two possible outcomes such as Spam or Not Spam, Cancer or No Cancer.
- Multinomial Logistic Regression: The target variable has three or more nominal categories such as predicting the type of Wine.
- Ordinal Logistic Regression: the target variable has three or more ordinal categories such as restaurant or product rating from 1 to 5.

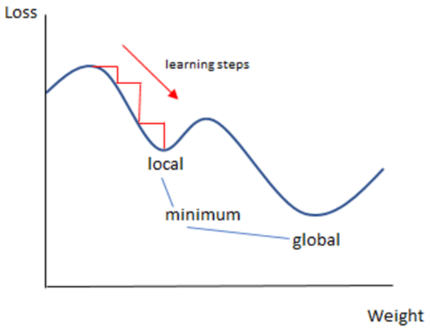
Our model comes under multinomial logistic regression as there are 3 categories positive , negative , neutral.

2. Stochastic Gradient Descent - Classifier (SGD-Classifier):

Stochastic gradient descent (SGD) computes the gradient using a single sample. SGD allows minibatch (online/out-of-core) learning. Therefore, it makes sense to use SGD for large scale problems where it's very efficient.

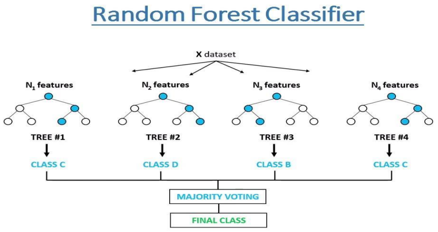
The minimum of the cost function of Logistic Regression cannot be calculated directly, so we try to minimize it via Stochastic Gradient Descent, also known as Online Gradient Descent. In this process we descend along the cost function towards its minimum for each training observation we encounter.

Another reason to use SGD Classifier is that SVM or logistic regression will not work if we cannot keep the record in RAM. However, SGD Classifier continues to work.



3. Random Forest Classifier

Random Forest is a bagging type of Decision Tree Algorithm that creates a number of decision trees from a randomly selected subset of the training set, collects the labels from these subsets and then averages the final prediction depending on the most number of times a label has been predicted out of all.

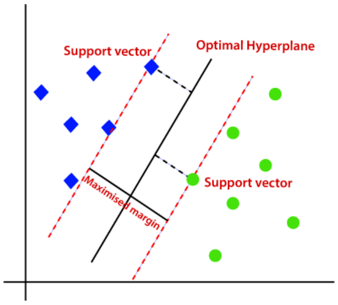


4. Support Vector Machine

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplanes.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.



5. Naive Bayes Classifier

Multinomial Naive Bayes algorithm is a probabilistic learning method that is mostly used in Natural Language Processing (NLP). The algorithm is based on the Bayes theorem and predicts the tag of a text such as a piece of email or newspaper article. It calculates the probability of each tag for a given sample and then gives the tag with the highest probability as output.

Naive Bayes classifier is a collection of many algorithms where all the algorithms share one common principle, and that is each feature being classified is not related to any other feature. The presence or absence of a feature does not affect the presence or absence of the other feature.

6. CatBoost

CatBoost is a high-performance open source library for gradient boosting on decision trees. It works well with multiple categories of data, such as audio, text, image including historical data.

It is a readymade classifier in scikit-learn's conventions terms that would deal with categorical features automatically. It can work with diverse data types to help solve a wide range of problems that businesses face today. It is developed by Yandex researchers and engineers, and is used for search, recommendation systems, personal assistant, self-driving cars, weather prediction and many other tasks. Also, it provides best-in-class accuracy.

7. Model performance:

Model can be evaluated by various metrics such as:

1. Accuracy-

Accuracy is defined as the percentage of correct predictions for the test data. It can be calculated easily by dividing the number of correct predictions by the number of total predictions.

$$\text{accuracy} = \frac{\text{correct predictions}}{\text{all predictions}}$$

2. **Confusion Matrix-**

   The confusion matrix is a table that summarizes how successful the classification modelis at predicting examples belonging to various classes. One axis of the confusion matrix is the label that the model predicted, and the other axis is the actual label.

3. **Precision/Recall-**

   Precision is defined as the fraction of relevant examples (true positives) among all of the examples which were predicted to belong in a certain class.

$$\text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

   Recall is defined as the fraction of examples which were predicted to belong to a class with respect to all of the examples that truly belong in the class.

$$\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

4. **F-1 Score**

   When using classification models in machine learning, a common metric that we use to assess the quality of the model is the F1 Score.

   This metric is calculated as:

   F1 Score = 2 * (Precision * Recall) / (Precision + Recall)

8. Conclusion:

That's it! We reached the end of our exercise.

Starting with loading the data so far we have done EDA , null values treatment, text pre-processing, encoding of categorical columns, word cloud, feature selection and then model building.

In all of these models our accuracy revolves in the range of 68% to 81% nearly.

And there is little bit of improvement in test accuracy score even after hyperparameter tuning. Similarly recall, precision and f-1 scores were also calculated for the models.

So the accuracy of our best model is 80.98% by SGD-classifier, which can be said to be good for this kind of dataset containing text for sentiment analysis. And multiclass classification results in good accuracy also.

9. References:

1. AnalyticalVidhya
2. Geeksforgeeks
3. Medium Blog
4. Kaggle
5. AlmaBetter Resources