

**PDPM IIIT DESIGN AND MANUFACTURING, JABALPUR**

**REPORT**

**IRRELEVANT Classification of Social Media Post During Disaster**

Pravesh vajpayee1, Sahil ahammed2, Munesh kushwah3

1Student, Discipline of Electronics and Communication Engineering, Indian Institute of Information Technology, Design and Manufacturing, Jabalpur, India, [2018194@iiitdmj.ac.in](mailto:2018194@iiitdmj.ac.in)

2Student, Discipline of Mechanical Engineering Indian Institute of Information Technology, Design and Manufacturing, Jabalpur, India, [2018216@iiitdmj.ac.in](mailto:2018216@iiitdmj.ac.in)

3Student, Discipline of Electronics and Communication Engineering, Indian Institute of Information Technology, Design and Manufacturing, Jabalpur, India, [2018342@iiitdmj.ac.in](mailto:2018342@iiitdmj.ac.in)

**Abstract :**

As one of the biggest online markets worldwide, 54.1% of internet users in India are active on social media . Facebook, instagram,twitter,youtube etc. are the most common social media in India, providing a platform to spread information throughout its user. However, fact-based information is not the only one circulated on the Internet.

The number of irrelevant posts shared throughout the Internet, especially social media, is concerning. Investigating irrelevant posts requires considerably longer time in collecting the data to compare. In addition, humans naturally are not very good at differentiating between relevant and irrelevant social media posts .

It makes machine learning become advantageous in dealing to this problem. However, the rapid changes of internet posts throughout time requires machine learning to be able to train its model dynamically. Incremental machine learning is proposed to solve this problem. We first introduce and compare different incremental learning techniques, and show that they are capable of producing better performance results.

**Introduction :**

Now, the incremental data classification method has been one of the key technologies of intelligent knowledge discovery. Compared with the traditional data classification strategies, incremental data classification has two advantages:

1) It can reduce the storage cost by discarding old samples;

2) The utilization of historical training results makes the successive learning faster.

As the work of professional bodies, volunteers, and others is increasingly mediated by computer technology, and more specifically by social media,1 research on crisis management has become more common . The emerging research field of crisis informatics has revealed interesting and important real-world uses for social media . Crisis informatics is ‘a multidisciplinary field combining computing and social science knowledge of disasters; its central tenet is that people use personal information and communication technology to respond to disasters in creative ways to cope with uncertainty’ .

During disasters and emergencies, it is necessary for emergency services to obtain a comprehensive situational overview for coordination efforts and decision making . In such situations, social media are increasingly used for the exchange of information while emergency services encounter the issue of information overload, amongst others .

Research indicates that supervised machine learning techniques are suitable for identifying relevant posts and filter out irrelevant posts, thus mitigating information overload . Besides the potential of improving the performance of algorithms for relevance classification, supervised machine learning techniques require a considerable amount of labeled data, which constitutes an issue due to the time-sensitive nature of disasters and emergencies . Furthermore, clear criteria for relevance classification are required, a usable interface to facilitate the labeling process and a mechanism to rapidly deploy retrained classifiers.

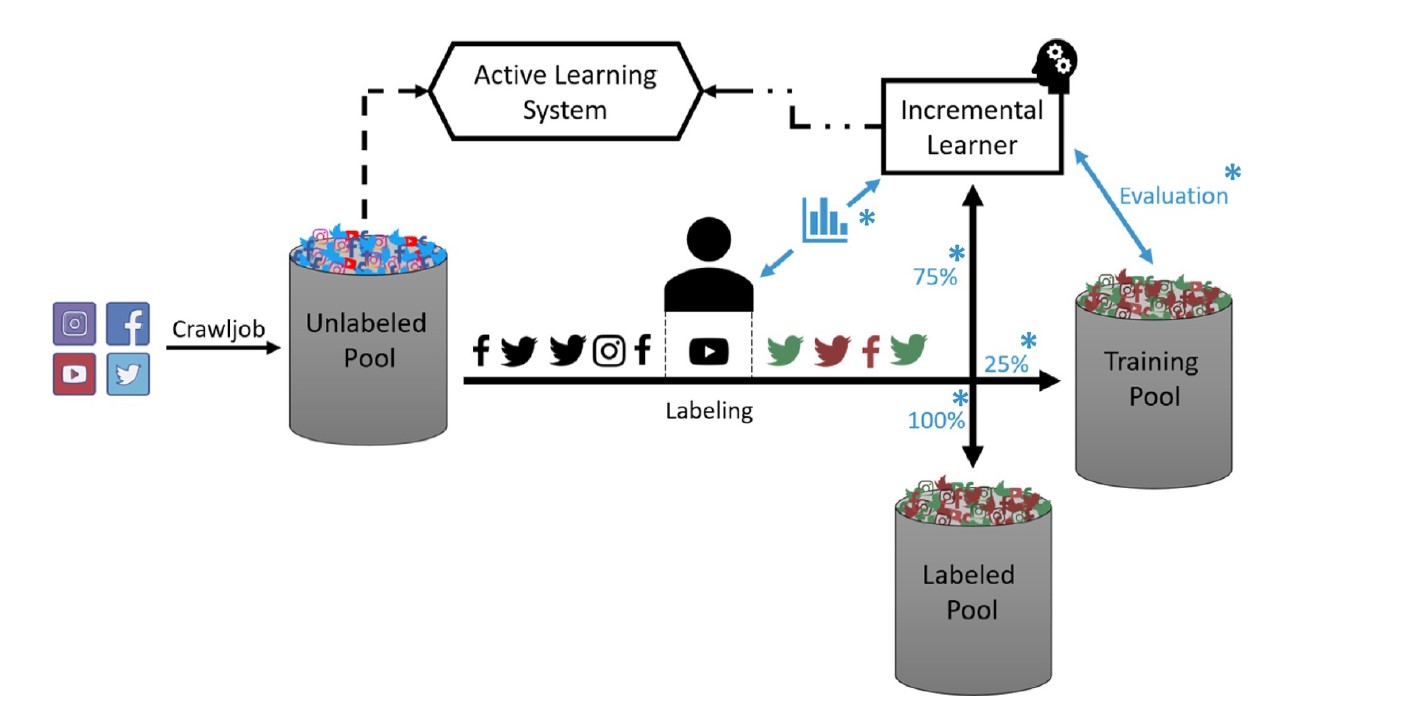
Based on a communication matrix and social media analytics framework, we designed and evaluated a system featuring active and online learning to support the information flow of integration of citizen-generated content featuring social media. Thereby, we seek to answer the following research questions:

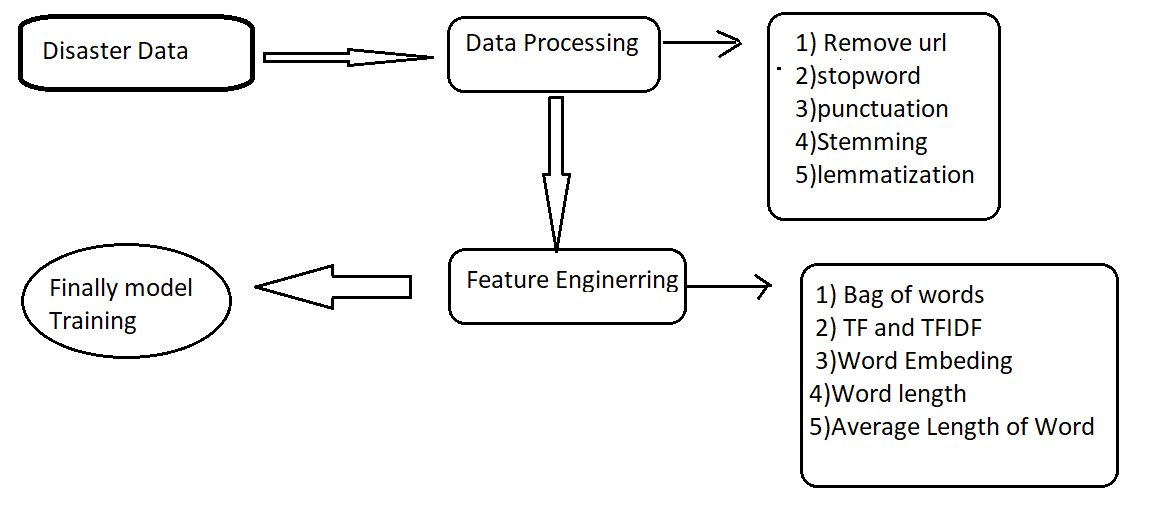
• What are suitable criteria for relevance classification and labeling in disasters and emergencies.

• How can existing supervised machine learning techniques for relevance classification be improved for use in real disaster and emergency environments.

• How can the amount of labeled data required for relevance classification be reduced by active incremental learning and transparent visualization of the classifier's quality.

• How can the dynamic retraining of relevance classifiers be supported by user feedback performance-wise using batch learning with feature subset selection.





**Data pre-processing :**

**a)Stopwords:**Stopwords are the most common words in any natural

language(a,an,the,you,can).

**b)Stemming:** is the process of reducing a word to its word stem that affixes to suffixes.

**c) Lemmatization** usually refers to doing things properly with the use of a vocabulary and morphological analysis of words,

**Feature Engineering :**

**a)Bag of words**  It creates a vocabulary of all the unique words occurring in all the

documents in the training set.

**b)TF :**Term frequency is the measurement of how frequently a term occurs within a

Document

**c)TFIDF:** Term frequency inverse document frequency it eliminates the the words

which is present in every documents.

**d)Word Embedding:**  It’s a conversion of word to Vectors and process (Word2vec

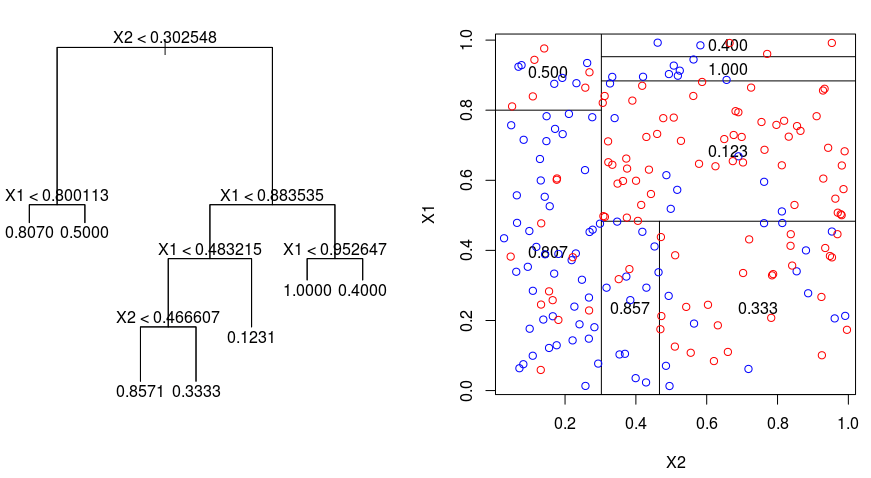
used is Twitter 50billion words)

**Approaches :**

We have approached this problem with three different solution :

**Incremental learning using Decision Tree**

**Decision tree :** A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

****

**Create the Decision tree classifier if not created.**

**Step 1:**

**Store the all the constraints of decision tree:**

i) Left and right Node

ii) Threshold

iii) leaf node value

iv)Take dominant class from of leaf node value.

**Step 2: Feature Detail : Detail of the features which includes**

i) min max and mean value if the feature is continuous.

ii) number of unique category if feature is category

**Step 3: Update feature detail: if new feature detail is introduced or there is change in**

**behaviour of feature.**

**Step 4 : Generate the data using an old classifier.**

i) start with the parent node generate random data using feature detail.

ii) go to the left node by using threshold Decision tree constraint.

iii) go to the right node by using threshold Decision tree constraint.

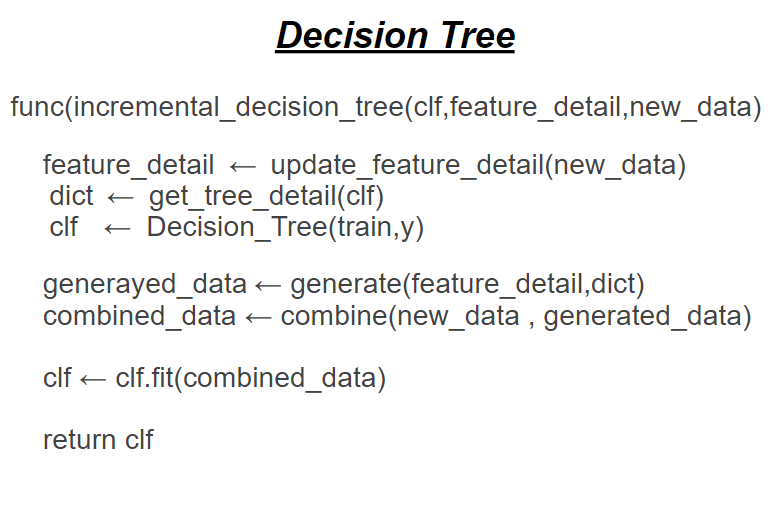
iv) traverse every node till reach to leave node.

v) Find Dominant class.

vi) Every time when node is traversed, generate data append to main data.

vii) Finally store dominant class of dominant class.

**Pseudo code for Decision Tree**

****

***Incremental Learning using SVMs***

The classification algorithm that is based on a support vector machine (SVM), are one of the good algorithms for incremental learning because of its theoretical properties and good empirical results. The theoretical basis of this algorithm is the classification equivalence of the SV set and the training set. In this algorithm, knowledge is accumulated in the process of incremental learning.

SVM theory is mainly derived from the problem of binary classification. Its main idea can be concluded as the following two points: First, it constructs a nonlinear kernel function to present an inner product of feature space, which corresponds to mapping the data from the input space into a possibly high-dimensional feature space by a nonlinear algorithm. Thus it is possible to analyze the nonlinear properties of samples in the feature space with linear algorithms. Secondly, it implements the structural risk minimization principle in statistical learning theory [3] by generalizing optimal hyper-plane with maximum margin between the two classes. Although intuitively simple, this idea actually plays the role of capacity controlling and makes the learned machine not only have small empirical risks, but also has good generalization performance. Therefore, SVM has many advantages in both theoretical base and practical prospect.

SVM algorithms use a set of mathematical functions that are defined as the kernel. The function of the kernel is to take data as input and transform it into the required form. Different SVM algorithms use different types of kernel functions. These functions can be different types. For example *linear, nonlinear, polynomial, radial basis function (RBF), and sigmoid.*

Introduce Kernel functions for sequence data, graphs, text, images, as well as vectors. The most used type of kernel function is RBF. Because it has localized and finite response along the entire x-axis.

The kernel functions return the inner product between two points in a suitable feature space. Thus by defining a notion of similarity, with little computational cost even in very high-dimensional spaces

In this approach I have divided data into three sets and trained three times and used the resultant .

**1)**We trained an SVM on TR1 (Training Set 1). Then we took the support vectors chosen from TR1, let us call these SV1 (Support vector ).

**2)** Then we have trained svm on TR2 .Then we have added and added them to SV1+TR2.

**3)**Again we ran the SVM training algorithm on SV 1 + TR2. At each step, we used the trained machine to classify TE . Now the SVs chosen from the training of SV 1 + TR2 were taken, let us all these SV 2, and TR3 was added to them and the resultant sv’s are stored in sv1 .

It can be extended for more data by training and testing again and again and storing the result . This incremental step will be repeated till all the data are trained .

**Pseudo code for SVM:**

TR=support\_vector\_ # resultant support vector

For x,y in datasets :

tr1=SVC( kernel) # defining the kernel

tr1.fit( X,y ) # training the data

sv=tr1.support\_vectors\_ # taking the support vector

TR=TR+sv # updating the result

**Incremental Learning using Partial fit method in Logistic Regression**

Logistic Regression statistical method is used for analyzing the dataset and produces a binary outcome. One or more autonomous variables may have consisted of the dataset. The result is determined by these variables that are dichotomous in nature. Which means only two results are possible . It is a specific category of regression and it is used in the best way to predict the binary and categorical output.

Logistical Regression method is used to regulate the impact of numerous autonomous variables which are conferred at the same time. This method also predicts any one of the two independent categories of variables. Logistic regression designs the best-fitting function with the help of the maximum likelihood method in order to maximize the probability of classifying the recognized data into the proper division . Various appliances of logistic regression are forecast market trends, to find the success and failure rates in results, the true or false category in recruiting employees based on their performance in need of employment in a company, image categorization, health care and analyze a group of people affected by Myocardial Infarction

Logistic Regression (aka logit, MaxEnt) classifier. In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the ‘multi\_class’ option is set to ‘ovr’, and uses the cross-entropy loss if the ‘multi\_class’ option is set to ‘multinomial’. (Currently the ‘multinomial’ option is supported only by the ‘lbfgs’, ‘sag’, ‘saga’ and ‘newton-cg’ solvers.)

This class implements regularized logistic regression using the ‘liblinear’ library, ‘newton-cg’, ‘sag’, ‘saga’ and ‘lbfgs’ solvers. Note that regularization is applied by default. It can handle both dense and sparse input. Use C-ordered arrays or CSR matrices containing 64-bit floats for optimal performance; any other input format will be converted (and copied).

The ‘newton-cg’, ‘sag’, and ‘lbfgs’ solvers support only L2 regularization with primal formulation, or no regularization. The ‘liblinear’ solver supports both L1 and L2 regularization, with a dual formulation only for the L2 penalty. The Elastic-Net regularization is only supported by the ‘saga’ solver

# Batch gradient descent with scikit learn (sklearn)

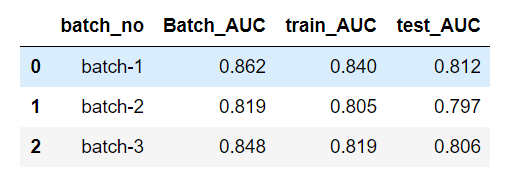
Batch learning means learning on the entire training set in one go, while what you describe is properly called mini batch learning. That's implemented in sklearn.liynear\_model.SGDClassifier, which fits a logistic regression model if you give it the option loss="log".

With SGDClassifier, like with LogisticRegression, there's no need to wrap the estimator in a OneVsRestClassifier -- both do one-vs-all training out of the box.

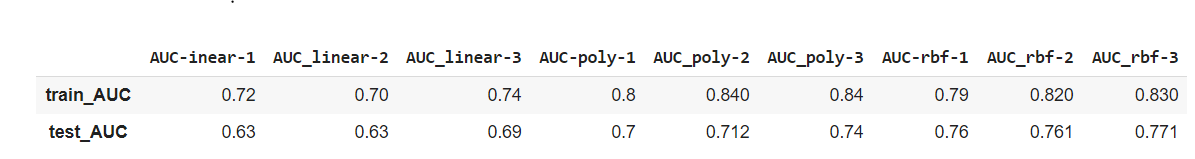
Then, to train on mini batches, I used the partial\_fit method instead of fit. The first time around, you have to feed it a list of classes because not all classes may be present in each minibatch:

**RESULT**:

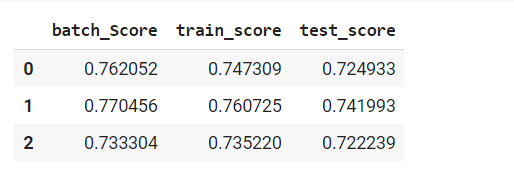
1. **Performance Decision Tree**

****

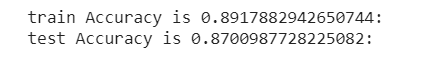
1. **SVM Performance**

****

1. **Logistic Regression**

****

Result by training on whole data -



**Conclusion :**

**a)Decision Tree:**

* The training AUC is improving after every batch which mean classifier is updating it's knowledge on train-data
* The training AUC in Batch\_3 is 0.819 and when the entire data is trained AUC of training data is 0.82,
* For the test\_data AUC in Batch\_3 is 0.806 and for entire\_data AUC is 0.804

AUC are nearly the same.

**B) incremental svm:**

* The training AUC is improving after every batch and it shows that svm is good for incremental training.
* For the test data AUC in batch\_3 is and for entire\_data is AUC is 0.841 which is good.
* The training AUC in batch is taking 2 sec and training whole data is 2.5 sec which is time efficient .
* From this we can conclude that incremental learning using svm can give good results and will be memory efficient .

**C)Logistic Regression :**

* The training AUC is improving after every batch which means the classifier is updating it's knowledge on train-data.
* The training AUC in Batch\_3 is 0.781 and when the entire data is trained AUC of training data is 0.891,
* For the test\_data AUC in Batch\_3 is 0.760 and for entire test data AUC is 0.870
* Thus we can say that incremental learning of logistic regression can give good and bad results.

.

**Reference** :

[1] Statista. Countries with the highest number of internet users as of june 2017.

[2] Hunt Allcott and Matthew Gentzkow. Social media and fake news in the 2016 election. Journal of Economic Perspectives, 31(2):211-236, 2017.

[3]V. Vapnik. The Nature of Statistical Learning Theory. Springer-Verlag, New York, 1995.

[4][Shuo Wang](https://ieeexplore.ieee.org/author/37696452600); [Jian-Jian Wang](https://ieeexplore.ieee.org/author/38195650900); [Xiang-Hui Gao](https://ieeexplore.ieee.org/author/37596005800); [Xue-Zheng Wang](https://ieeexplore.ieee.org/author/37900822200) Pool-based active learning

based on incremental decision tree