**SPL-1: PROJECT REPORT, 2019**

**Bug Severity Classification**

**Using Decision Tree**

**SE-305 : Software Project Lab-1**

**Submitted by:**

**Nazmul Hossain**

**BSSE Roll No. : 1032**

**Session: 2017-18**

**Supervised by**

***Dr. Mohammad Shoyaib***

**Designation: Professor**

**Institute of Information Technology**



**Institute of Information Technology**

**University of Dhaka**

Table of Contents

1. Introduction………………………………………………………………………………....3

1.1. Background Study……………………………………………………………………...3

1.2. Challenges ……………………………………………………………………………...5

2. Project Overview …………………………………………………………………………...5

3. User Manual………………………………………………………………………………...8

4. Conclusion ………………………………………………………………………………….9

Reference …………………………………………………………………………………… 10

**1. Introduction**

This project targets the problem of identifying the severity of Bug Reports.

It is very difficult for a person himself to identify the bug severity analysing the whole bug

reports of a software. For this purpose, this project aims to find a bug tracking system to

analysis bug reports using the techniques of Natural Language Processing and predict

severity of bugs in a Machine Language approach.

* 1. **Background Study**

**1.1.1.Natural Language Processing**

Natural language processing (NLP) is a branch of artificial intelligence that helps computers

understand, interpret and manipulate human language. I have studied about following

techniques of NLP .

**Tokenization:** Tokenization is the process of splitting the text into smaller parts called

tokens. I have studied how to tokenize the content of a text file.

**Stop Word Removal:** In natural language processing, useless words (data), are referred to as

stop words.Stopword filtering is a common step in preprocessing text for various purposes.

We would not want these words taking up space in our database, or taking up valuable

processing time. For this, we need to remove them.

**Stemming:** Stemming is the process of reducing inflected (or sometimes derived) words to

their word stem , base or root form. Stemming is important while preprocessing a text. An

well-known stemmer for this purpose is “Porter Stemmer”. The Porter stemming algorithm is

a process for removing the commoner morphological and inflexional endings from words in

English.

**1.1.2. Machine Learning**

**Feature Selection:** Feature selection in machine learning refers to the process of isolating

only those variables (or features) in a dataset that are pertinent to the analysis. To select

feature effectively is very much important for any kind of analysis. In this project, I have

selected feature in a very simple way(though not effectively). The tokens of highest

frequencies have been selected as the features here. The number of features depends on the

size of dataset.

**Classifier Implementation:** In machine learning, classification is the problem of identifying

to which of a set of categories (sub-populations) a new observation belongs, on the basis of a

3

training set of data containing observations (or instances) whose category membership is

known. An algorithm that implements classification, especially in a concrete implementation,

is known as a classifier. There are many types of classifier (Decision tree, Random Forest

tree, Naive Bayes, KNN etc), among them I have chose decision tree to implement in this

project.

**Decision Tree :** Decision tree builds classification or regression models in the form of a tree

structure. It breaks down a dataset into smaller and smaller subsets while at the same time an

associated decision tree is incrementally developed. The final result is a Tree with decision

nodes and leaf nodes. A decision node has two(in binary tree) or more branches . Leaf node

represents a classification or decision. The topmost decision node in a tree which corresponds

to the best predictor called root node. Decision trees can handle both categorical and

numerical data.

**Information Gain :** Information gain (IG) measures how much ‘information’ a feature

gives us about the class. Features that perfectly partition should give maximal information.

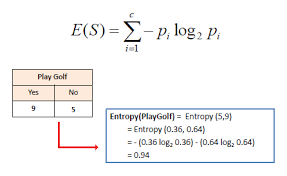
Unrelated features should give no information.

Information Gain = Entropy(parent) – [average entropy(children)]

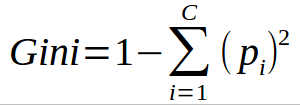
**Entropy :** In machine learning, Entropy is a measure of the randomness in the information

being processed. The higher the entropy, the harder it is to draw any conclusions from that

information.



**Gini Index :** In machine learning, the Gini is a measurement of impurity.



where Pi denotes the proportion of instances belonging to class i (i = 1, … , c).

**1.2. Challenges**

I have come across several challenges during my project. Some of those are as follow

● The main challenge of this project was to learn machine learning and implement bug

tracking technique.

● It took a lot time to realize the ID3 algorithm for decision tree,which employs a

top-down approach.

● I had to modify the main ID3 algorithm according to the requirements of dataset. I

had to generate the decision tree from numeric data and represent it as a binary tree.

● I had to try again and again to optimize the code for a satisfactory execution time &

memory limit.

● It was difficult to get a satisfactory accuracy.

● I implemented Gini index as well as Entropy based calculation to see which gives

more better result.

● I had to select competent features and generate the numeric dataset for the classifier.

● I had to learn some Natural Language Processing techniques & implement the

procedure of tokenization, parsing, stopword removal.

● I had to bother about reading the huge reports as dataset through machine.

**2. Project Overview**

The whole project has been completed through 2 major steps. The first is “Dataset

Processing” and the second is “Decision Making”. Both of the two are consist of several

procedures.

**2.1. Dataset Processing**

Firstly, the classified dataset(bug report or others) had to be analysed through a sequential

procedure. The procedures that the project follows are given below:

**Open File and Read from file :** Firstly the file has been opened in a stream. Then the

file(where exist the classified bug reports of a software,such that Mozilla Firefox,GCC,

Eclipse) has been read line by line from the stream, every line here refers to a bug report.

Then I had to consider only the summary of the bug report & the class it refers to.

**Clean Up :** Each bug report has to be cleaned up, here clean up means, without words, the

punctuation marks,numbers should be cleaned up & replaced by whitespace.

**Tokenization :** Then every bug report has to be tokenized & store in vector. The vector of

these vectors(that contain the token of each bug report) represents the whole dataset.

**Stop Word Removal :** If there exists stop-words, those need to be removed from the tokens.

So I had to match the stop-word list(in English) with the tokens & if any stop word is found

it was removed.

**Stemming :** Porter Stemmer was used for stemming, so that the tokens are in base form.

**Frequency Count :** After the preprocessing, the frequency of every token is counted both as

per bug report & the whole dataset.

**Feature Selection :** The tokens of highest frequency are selected as the features. The number

of features depends on the size of dataset. Normally I have selected thousands features for

larger dataset(above 1500 KB), & five hundred features for smaller dataset.

**Generate Dataset & Create File :** The dataset would be generated with the word frequency.

The number of the column will be same as the number of features,the column heading will be

named after the feature name. The number of rows will be same as the number of bug report

in the dataset. Each of the cell will contain the frequency of corresponding feature in the

respected bug report. Then a file will be created in output file stream & the numeric data will

be inserted into it. At this point,our desired dataset is ready. Now it should be analysed so that a newly inserted bug report can be classified.

**2.2. Decision Making**

At this portion, the classifier has been implemented using previously generated data. This

stage follows the following procedures.

**Read File & Store the Table :** Read the file from input stream line by line(per bug report),

store the info of frequency per feature & the respected class. The data structure I have used in

this purpose is the vector of the structures.

. **Select Training Data :** 90% data from each class has been selected as training data

randomly. These training data would be used to implement classifier.

**Select Testing Data :** Remaining 10% data has been used as the test data for testing the

result.

**Classifier Implementation :** Then using the training data the classifier has to be

implemented. I have implemented decision tree as a classifier using the self modified version

of ID3 algorithm. This algorithm use information gain to classify. I have calculated

Information Gain in two ways, using Entropy based calculation & Gini Index.

**Classifier Implementation :** Then using the training data the classifier has to be

implemented. I have implemented decision tree as a classifier using the self modified version

of ID3 algorithm. This algorithm use information gain to classify. I have calculated

Information Gain in two ways, using Entropy based calculation.

**Calculating Information Gain :** Information gain (IG) measures how much “information” a

feature gives us about the class. In this project, I have calculated information gain using

procedures of impurity measurement, Entropy.

**Entropy Based Calculation:** Entropy is measured by summation of the negated product of a

class probability & 2-based log probability of the attributes. Information gain is measured by

subtracting the parent entropy by the average of child entropy.

**Make Tree Recursively :**

A recursive function every time calculate the total entropy(in Entropy based calculation) or

Gini(in Gini index based calculation) of the table, if the entropy or gini is 0 (actually close to

zero),that means the information gain is maximum, then the leaf is found and the most

frequently occuring class in the dataset will be the decision of that leaf. If total entropy or

table gini is greater than 0 (not close to zero), then we have to found a feature as the node.

For this the dataset will be splitted into two parts(as we wanted to generate binary tree) with

respect to each element & the the information gain will be calculated. For which attribute,the

information gain is maximum,that corresponding feature will be selected as new node & the

whole data set will be splitted into two parts with respect to that attributes. Then the recursive

function will be called again with the left sided dataset & the right sided dataset.

This procedure will be going on recursively until the every leaf is found. Each time,we find a

leaf or a node.

**Test the Tree with Testing data :** The class of the previously selected (randomly) testing

data will be predicted using the decision tree.

**Accuracy Measurement :** The predicted classes will be compared with the actual classes of

the testing data. Then accuracy will be calculated in percentage.

**3. User Manual :**

● The user need to have a python compiler.

● Download the datasetProcessing.cpp file. If the user has a Bug Report Summary of a

software that is previously classified, then run the executable file with the data file as

input. Then numeric data will be generated.

● If one have a ready numeric dataset then he/she only need to build & run the

textProcessing.py file with the corresponding dataset. Then the test dataset will be

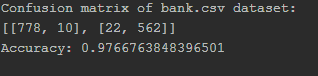
classified & he/she can also see the Accuracy of class predction.

3.1. Output

For using iris.csv file my output is

Capture1.PNG

Again ,for using bank.csv file my output is:



**4. Conclusion :**

The approaches in this program are efficient enough to classify small dataset within the range

(80-95)% accuracy. But while dealing with larger dataset, the required execution time is

above 60seconds or above.

**5. References :**

**1.** [https://machinelearningmastery.com](https://machinelearningmastery.com/)

2. <https://www.datacamp.com/community/tutorials/text-analytics-beginners-nltk>

3. Severity Prediction of Software Bug, Ahmed Fawzi Otoom , Information and

Communication Systems (ICICS), 2016

4. <https://towardsdatascience.com/entropy-how-decision-trees-make-decisions-2946b9c18c8>

5. <https://www.geeksforgeeks.org/decision-tree-introduction-example/>

6. <https://www.brighthubpm.com/project-planning/105851-how-to-create-a-decision-tree/>

7. <https://medium.com/@datamonsters/text-preprocessing-in-python-steps-tools-and-examples-bf025f872908>

8. <https://github.com/Nazmul1932/SPL1>