IBM IGNITE Defect Classify accelerates test using machine intelligence

Real-world experience in Defect Classification

Abhishek Mitra
Ilan S Jayaraman
Andrew Williams
Global Business Services, IBM

Abstract:

- The IGNITE Defect Classify application by IBM uses complex cognitive approaches to reduce defect turnaround.
- This document outlines an original approach to Defect Classification initiated within IGNITE Defect Classify application and how it instantly derives defect classifications based on historic records of past classifications. It describes the machine learning approach, text preprocessing and modeling used to transform and benefit typical engagements, moving them from a defect detection experience to defect prevention.

Introduction

"Test less and test right" has been a mantra within the IBM Quality and Testing service area for a few years now. When applied to client-focused solutions, this mantra can dramatically transform client perspective from traditional defect detection, to a defect prevention and prediction methodology. This paper outlines one solution that uses this as part of its ethos.

What is IGNITE Defect Classify?

Traditionally, defects have always been treated in a reactive manner. Any incorrect classification may result in increased turnaround in resolving the defect that can impact the time to market for the application under test. IGNITE Defect Classify is an IBM cognitive solution that changes this perspective by providing real-time classification on the origin of the defects based on existing defect patterns. This will improve user identification of the root cause of the defect, resulting in quick referral of the defect to the correct resolving team.

What problem does this solve?

Sometimes when a defect is raised, it is assigned to the incorrect development team or resolver based on an incorrect assumption regarding the origin of the defect. Is it code-related? Is it related to an environment? The assigners use their own insight or application experience to direct the defect to the group they assume can resolve it correctly.

If incorrectly assigned, the new defect—potentially with a high severity or impact—will sit at the bottom of someone's in-tray awaiting analysis and eventual reversion to the originator before the defect is reassigned, hopefully to the correct responder. An incorrect defect assignment can delay resolution time by hours, or even days.

The ability to correctly assign a defect based on automated analysis of the underlying patterns of defects will enable the raiser to quickly take preventive actions. This helps to ensure that defects are being addressed by the most appropriate team, thereby, eliminating any cycle time.

IGNITE Defect Classify reduces defect turnaround time

Consider the case of a tester from a global insurance client who has been raising and assigning defects to coders, environment specialists, or business analysts, based on personal interpretation of the defects. After two days of analysis, coders suggest that the defect is an environment issue and the defect should be reassigned to the environment specialists. Then after correct reassignment, the defect is fixed in a couple of hours. This situation is far too common.

If the tester had the ability to correctly identify the origin of defects and assign the defect to the right team from the start, there would be no time delay.

IGNITE Defect Classify addresses this situation and provides the ability to do the following:

- Cognitively identify the origin of the defect in real-time
- Substantiate its classification recommendation by providing confidence levels
- Override the classification for enabling machine learning based on the feedback provided

The IGNITE Defect Classify engine thinks and learns

IGNITE Defect Classify uses intelligent classification,[2] a branch of machine learning which classifies and prioritizes incoming defect tickets with minimal to no human involvement. The approach utilizes machine learning to identify how past tickets have been classified and routed, based on patterns and past actions of human service agents. From this information, it creates a data-driven model that automatically classifies new incoming tickets.
Here is an example of the processing workflow, which describes how a defect ticket is processed to be classified as either a code, or no-code type of defect, using a pre-trained Random Forest model.[1]

Dataset description:
A training dataset is created based on a set of closed defect tickets text describing summary and detailed description of defects and resolution details. Corresponding class labels (e.g., Code or No-Code) are provided by expert classification of the tickets based on resolution details. A large volume of classified defects is required for this step.

Text preprocessing:
Text preprocessing includes the removal of missing data (e.g., tickets with no defect summary or output labels). In addition, English stop words (e.g., “but”, “again”, “there”, “about” and “once”) and domain-specific stop words (e.g., test executers name/email ids/phone numbers, etc) are removed. The output of test preprocessing is a cleaned dataset, which contains keywords defining a defect category.

Predicting defect classification with speed, accuracy and intelligence
As shown in Figure 1, a pre-trained Random Forest model is used within IGNITE Defect Classify to predict the most probable or likely defect class. The Random Forest model is one of the most popular decision tree-based ensemble models.

In the figure, we can assume that the colors red and blue represent defect tickets with Code and No-Code type of defects respectively (in feature space). The Random Forest model learns how to draw a good decision boundary between Code and No-Code type of defects during its training phase. When new defect details are proposed, the classifier predicts its class by finding the defects position with respect to the already learned decision boundary.

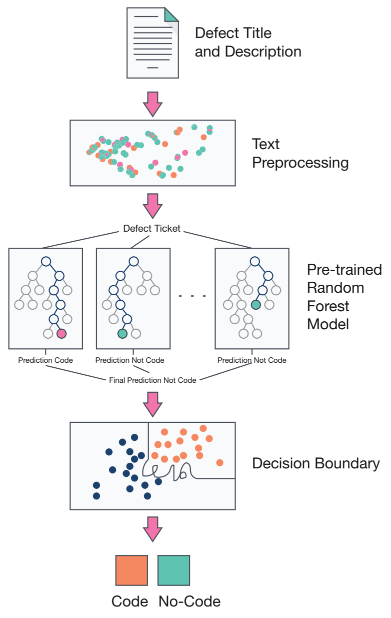

Figure 1. Defect classification using Random Forest classifier for Code and No-Code types binary defect classification

Training the defect classifier within IGNITE Defect Classify

As part of IGNITE Defect Classify, the defect classifiers need to be trained in the following phases:

1. Training phase: Train a Random Forest algorithm using a dataset comprised of the set of text documents (e.g., software defects) and their corresponding labels
2. Prediction phase: Utilize the trained model to predict labels of unseen defects.
3. Feature extraction: Perform Term Frequency/Inverse Document Frequency vectorization to extract numeric features from each preprocessed text document that is used to train Random Forest classifier.

To simplify the problem–in the given example–IGNITE Defect Classify uses binary classification (e.g., Code/No-Code). However, in actual case, our defect ticket dataset comprises of more than two labels, e.g., code/design/requirement/test-process, which allow a multi-class classification problem to be handled by our classification engine, which is finally deployed in the cognitive system.

To account for open and closed defects, a set of models are trained with only a "defect title" and "defect description" to be used for the following:

- Prediction of class label for open defect
- Another set of models trained with an additional input called "resolution details" to be used for predict-the-class label for closed defect

In essence, this training is the part of IGNITE Defect Classify which intelligently consumes existing defect patterns, and then uses them and other models to recommend where the next defect should be sent for resolution. This reduces opportunities for initial misdirection of a defect and reduces the overall defect turnaround and defect mitigation effort.

Interpreting prediction results

As part of IGNITE Defect Classify, the defect classifiers outcome should be interpreted to extract the predicted classification of the defects.

Here is a sample output when a new defect is input to our "defect class" classifier:
[{"defectClass": "test process", "accuracy": 65}, {"defectClass": "data", "accuracy": 16}, {"defectClass": "environment", "accuracy": 11}, {"defectClass": "requirements", "accuracy": 8}]

As you can see, the output from the classifier is a JSON containing the probability score of the defect falling into a particular "defect class" (e.g., "test process" has the highest probability in this example).

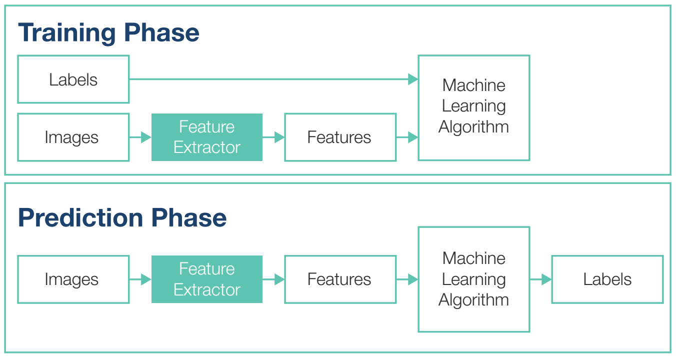

Figure 2. Classifier Training and Testing Phase

Conclusion

By implementing cognitive defect classify techniques in the test process:

- Defect turnaround time is reduced significantly
- Correct preventive actions based on defect pattern analysis are taken

At its heart, IGNITE Defect Classify quickly gives the users cognitively-derived advice on the origin and cause of the defect, ensuring the correct resolver is assigned. This reduces the defect turnaround time and defect mitigation effort by using a Natural Language Processor, intelligent classification and the enterprises defect history. This will also improve the overall delivery time for new and enhanced business applications.

References

[1] Research of Text Categorization Model based on Random Forests, IEEE International Conference on Computational Intelligence and Communication Technology (CICT), February 2015, http://ieeexplore.ieee.org/document/7078689/?reload=true

[2] Algorithms for Text Categorization: A Comparative Study, World Applied Sciences Journal 22 (9): 1232-1240, 2013, https://pdfs.semanticscholar.org/b04b/a34d802f4e3f176ad0a8015fccb6ff98addd.pdf

21013421-USEN-00