CREAM Classification of Incident Reports by Machine Learning

#### Created by Ka Lai (Karl) Yung in collaboration with Caroline Morais and Edoardo Patelli

### Program Overview

This collection of MatLab code reads and predicts the CREAM (Cognitive Reliability and Error Analysis Method) categorization of incident reports using machine learning. By using U.S. Chemical Safety and Hazard Investigation Board (CSB) reports, the code currently produces approximately 85 ~ 91% accurate categorization.

Essentially, the process begins with the MatLab file "getImportantSection.m". In the file's current form, it targets the recommendation, lessons learned, and advice sections of incident report PDFs. Using a confidence scoring system, the most likely start and end of a target section is found, and the section is output to later code.

The second file "dataCompiler.m" is responsible for aggregating all the information needed for machine learning. Using another confidence scoring system, the code takes the PDF file name and finds the most likely corresponding entry in an Excel spreadsheet. This spreadsheet is the manually categorized data from a researcher, and this gives the code the likely correct categories for each incident in the PDFs. The output is a combination of target section text and known, correct categorization of it.



Diagram of the overall data flows (simplified)

The last file, "reportClassifier.m", trains and tests the SVM (Support Vector Machine) using the data input from the previous two files. The section texts are converted into bagOfWords MatLab objects as X. The category data is optionally simplified and serves as the Y. The code partitions the X and Y data into a training set (90% of total) and a testing set (10% of total). For each CREAM category, a SVM model is trained using training X and Y sets, then it is tested using the testing X and Y sets. At the same time, run information is recorded and an overall accuracy of all test sets in all categories is calculated.

### Program Prerequisites

MatLab text analytics toolbox: used for extracting text strings from PDFs in dataCompiler.m

MatLab statistics and machine learning toolbox: used for SVM functions in reportClassifier.m

Human factor incident report PDFs: used as inputs for machine learning

Manual CREAM categorization spreadsheet: used as desired outputs for machine learning

### Run Instructions

1. Ensure dataCompiler.m, getImportantSection.m, reportClassifier.m, preprocess.m, and ignoreWords.csv are in the same folder that is the current MatLab workspace along with Classification Table V9 (new statistics).xlsx and the folder with report PDF files. It is suggested that a few dozens of U.S. Chemical Safety and Hazard Investigation Board (CSB) Case Studies and Investigation Reports or National Transportation Safety Board Accident Reports are used.
2. Ensure the text analytics and statistics & machine learning toolboxes are installed.
3. Adjust the settings in reportClassifier.m (it is suggested to have all boolean settings to be true) and run this MatLab file. For results, see the total accuracy in the command window and testing details in the debug array.

### Possible Expansions

Additional Incident Report PDFs

The current collection of reports comes from different organizations with drastically different formats and vocabularies. The formats range from a few concise pages in CSB reports to a 200-page letter to the US president on the BP oil spill.

Since CSB reports have shown decent results, it is suggested that additional CSB reports can be added to improve the robustness of the trained model. The same suggestion can be applied for National Transportation Safety Board (US) reports as its collection is currently lacking in numbers.

Replacing Less Reliable MatLab Functions

Currently, reading PDF files is handled by the MatLab text analytics toolbox function "ExtactFileText". Although this function successfully parses the entirety of all current reports, it sometimes misidentifies the location of formatted section headings. In addition, it does not identify the texts' fonts and text sizes. These issues reduce the potential accuracy of the final results and can be solved by a better PDF file parse function.