Project Update 2 Report
CSC-4220/5220
Dr. Akond Rahman
Andrew P. Worley, Raunak S. Hakya, & Jonathan A. Gibson

Objective:

The goal for the second update regarding the CSC-4220/5220 project was to implement 5 different classification algorithms across 4 different training/testing models. This gave us 20 different result sets, one for each unique model-classifier pair.

The steps to achieve this goal are:

- Take CSV dataset file as input and separate out the independent data columns(s) with the dependent column being SECU_FLAG.
- Apply Decision Tree, Random Forest, ANN (Artificial Neural Network), kNN (k Nearest Neighbor), & Naïve Bayes classification.
- Apply 10-fold cross validation and report prediction accuracy using precision, recall, and F-measure.

Dataset Description:

The original shape of our data set is 300056 (rows) and 12 (columns).
The given columns are:

- REPO: The name of the repository.
- HASH: The commit hash.
- TIME: The commit timestamp.
- ADD_LOC: The quantity of lines (across the commit files) where content was added.
- DEL_LOC: The quantity of lines (across the commit files) where content was deleted.
- TOT_LOC: The sum of ADD_LOC and DEL_LOC.
- DEV_EXP: A numeric expressing total contribution experience of the commit developer.
- DEV_RECENT: A numeric total expressing the recent contribution experience of the commit developer.
- PRIOR_AGE: The number of commits representing the age of the repository.
- CHANGE_FILE_CNT: The quantity of files that were changed for the commit.
- SECU_FLAG: The security status (NEUTRAL or INSECURE).
- REPO_TYPE: The type of content contained in the repository.

The dataset was used to generate two components for each training model. The components consisted of a single column array for the dependent variable (Y, the target), in our case this was the security status of the repository (SECU_FLAG), while the independent variable columns (X, the model data) were collected together into a 2D matrix. This 2D matrix was comprised of one or more columns from the dataset, based on the model type, shown in "Model Descriptions" below. Additionally, there were two columns that needed to be encoded for input into the sci-kit learn classification functions, SECU_FLAG and REPO_TYPE. These columns contained 2 and 5 unique data strings respectively, but the sci-kit learn functions required numerical input. Each column was simply enumerated based on each unique value. While the SECU_FLAG column was duplicated with its encoded values into the 13th column TARGET, the REPO_TYPE column was just replaced by its encoded equivalent. Detailed information can be found in "Dataset Statistics & Encoding" below.

For the execution of the learning algorithms, a combination of `cross_val_predict()` and `cross_val_score()` were used to record the model's predictions and how it reflected the true dataset target values. The resulting predictions were then sent to `metric.classifcation_report()` for calculation of precision, recall, and F-score. Although using two different cross validation functions made our program less efficient, we were able to confirm accuracy measurements from the classification report and the average of the testing phase across 10 folds. The true positive, false negative, false positive and true positive values were calculated using the ground truth (correct) target values and the estimated targets as returned by the classifiers.

Dataset Statistics & Encoding:

Encoded Targets:

NEUTRAL –> Encoded as: 0, Count: 276260 / 300056, Percentile: 92.0%
INSECURE –> Encoded as: 1, Count: 23796 / 300056, Percentile: 8.0%

Encoded Repository Types:

ComputationalChemistry –> Encoded as: 0, Count: 52395 / 300056, Percentile: 17.5%
Astronomy –> Encoded as: 1, Count: 67344 / 300056, Percentile: 22.4%
ComputationalBiology –> Encoded as: 2, Count: 174933 / 300056, Percentile: 58.3%
ComputationalScience –> Encoded as: 3, Count: 3137 / 300056, Percentile: 1.0%
ComputationalIntelligence –> Encoded as: 4, Count: 2247 / 300056, Percentile: 0.7%

Model Descriptions:

The 2D matrix for the model, mentioned above, consisted of one or more data columns, based on the model type. We used 4 different model types: Type, Size, Time, and Full. The data columns used for these models is shown below:

Type-based: ['REPO_TYPE']
Size-based: ['ADD_LOC', 'DEL_LOC', 'TOT_LOC']
Time-based: ['PRIOR_AGE']
Full: ['ADD_LOC', 'DEL_LOC', 'TOT_LOC', 'PRIOR_AGE', 'REPO_TYPE']

Results of Cross-fold Validation:

NOTE: We used 50 decision trees as a parameter for our random forest classifier, 3 as the neighbors parameter/threshold for the kNN classifier, and the Gaussian variant for our naïve bayes classifier. Additionally, the values for the precision, recall, and F-score, from the classification report, are for both the NEUTRAL and INSECURE targets, represented by (NEUTRAL, INSECURE). The accuracy measure listed is the median for each model-classifier pair; in many cases the median is better at representing the values in a list because it doesn't skew results based on anomalies, like an average/mean does.

<u>The Type-Based Model:</u> ['REPO_TYPE']

Metric / Classifier	Decision Tree	Random Forest	ANN	kNN	Naïve Bayes
Accuracy	0.9207	0.9207	0.9207	0.9207	0.9207
Precision	(0.9199, 0.0000)	(0.9199, 0.0000)	(0.9207, 0.0000)	(0.9199, 0.0000)	(0.9207, 0.0000)
Recall	(0.9890, 0.0000)	(0.9890, 0.0000)	(1.0000, 0.0000)	(0.9890, 0.0000)	(1.0000, 0.0000)
F-Score	(0.9532, 0.0000)	(0.9532, 0.0000)	(0.9587, 0.0000)	(0.9532, 0.0000)	(0.9587, 0.0000)
True Negatives	273222	273222	276260	273222	276260

False Negatives	3038	3038	0	3038	0
False Positives	23796	23796	23796	23796	23796
True Positives	0	0	0	0	0

The type-based model was not very useful as no positives were found. We believe this is due in part to the small number of positives in the training dataset and the limited variation of the independent variable. For the ignorant eye, these results may seem acceptable, but given that the purpose is to detect both NEUTRAL and INSECURE instances, the REPO_TYPE column by itself is a very poor model to use for this particular dataset.

<u>The Size-Based Model:</u> ['ADD_LOC', 'DEL_LOC', 'TOT_LOC']

Metric / Classifier	Decision Tree	Random Forest	ANN	kNN	Naïve Bayes
Accuracy	0.9092	0.9161	0.9207	0.9072	0.0864
Precision	(0.9208, 0.0846)	(0.9207, 0.0813)	(0.9206, 0.0000)	(0.9209, 0.0904)	(0.9313 0.0794)
Recall	(0.9843, 0.0169)	(0.9931, 0.0071)	(0.9960, 0.0000)	(0.9841, 0.0183)	(0.0087, 0.9925)
F-Score	(0.9515, 0.0282)	(0.9555, 0.0131)	(0.9568, 0.0000)	(0.9515, 0.0305)	(0.0173 0.1470)
True Negatives	271912	274349	275153	271875	2413
False Negatives	4348	1911	1107	4385	273847
False Positives	23394	23627	23720	23360	178
True Positives	402	169	76	436	23618

The size-based model worked better than the type-based model overall, though still not great. We believe the greater quantity of data columns allowed a closer fit; however, the results are not significantly different from each other. The results from the naïve bayes classifier were abysmal for the size-based model. This is because the naïve bayes classifier makes a large, initial assumption about the independence of the data columns. For this dataset, many of the data columns have innate dependency because of the relationship between the properties of commits. This resulted in very low accuracy, recall, and F-score for this model.

<u>The Time-Based Model:</u> ['PRIOR_AGE']

Metric / CLassifier	Decision Tree	Random Forest	ANN	kNN	Naïve Bayes
Accuracy	0.9157	0.9147	0.9207	0.8885	0.9207
Precision	(0.9179, 0.0039)	(0.9179, 0.0052)	(0.9207, 0.0000)	(0.9169, 0.0248)	(0.9207, 0.0000)
Recall	(0.9611, 0.0018)	(0.9605, 0.0024)	(1.0000, 0.0000)	(0.9318, 0.0201)	(1.0000, 0.0000)
F-Score	(0.9390, 0.0024)	(0.9387, 0.0033)	(0.9587, 0.0000)	(0.9243, 0.0222)	(0.9587, 0.0000)

True Negatives	265500	265337	276260	257431	276260
False Negatives	10760	10923	0	18829	0
False Positives	23754	23739	23796	23317	23796
True Positives	42	57	0	479	0

The time-based model was not very successful. The decision tree and kNN algorithms did perform better when using the type-based model, but by an insignificant value. We believe this is in part due to an opposite problem, as the independent variable here has a wide range of values. Although the larger range of values should technically give more information for the classifier, this kind of distribution of values can "confuse" the model and not allow the model to settle on definitive rules or high enough probabilities for proper classification.

<u>The Full Model:</u> ['ADD_LOC', 'DEL_LOC', 'TOT_LOC', 'PRIOR_AGE', 'REPO_TYPE']

Metric / Classifier	Decision Tree	Random Forest	ANN	kNN	Naïve Bayes
Accuracy	0.8130	0.8771	0.9207	0.8891	0.0871
Precision	(0.9141, 0.0408)	(0.9153, 0.0215)	(0.9207, 0.0192)	(0.9175, 0.0314)	(0.8480, 0.0783)
Recall	(0.8484, 0.0749)	(0.9092, 0.0232)	(0.9998, 0.0000)	(0.9333, 0.0251)	(0.0124, 0.9741)
F-Score	(0.8800, 0.0529)	(0.9122, 0.0223)	(0.9586, 0.0001)	(0.9253, 0.0279)	(0.0245, 0.1450)
True Negatives	234380	251179	276209	257840	3436
False Negatives	41880	25081	51	18420	272824
False Positives	22013	23244	23795	23198	616
True Positives	1783	552	1	598	23180

The full model worked the best for all the algorithms for detecting the INSECURE commits, though still not acceptable. Once more, we believe this is the result of the lack of positives in the dataset. It also took significantly longer to run, due to the multiple independent variables to process. Again, the naïve bayes was the best in terms of INSECURE F-Score and demonstrating a high recall. Unfortunately, the naïve bayes accuracy increase was miniscule compared to that of the size-based model.

Classification Time:

Based on the relative complexity of the classifiers, some of the classifiers took longer than others. Although the absolute values can be taken with a grain of salt (because we doubled our program's workload by using 2 separate cross validation functions), the ratio of complexity to time required, among all the classifiers should be accurate.

Decision Tree: 1.00 minute(s) and 15.66 seconds
Random Forest: 20.00 minute(s) and 23.87 seconds
ANN: 186.00 minute(s) and 5.38 seconds
kNN: 30.00 minute(s) and 7.44 seconds
Naïve Bayes: 0.00 minute(s) and 28.77 seconds

Summary:

Unsurprisingly, the results for the decision tree and the random forest classifiers were very similar (because a random forest is just a collection of decision trees). Although both classifiers attempted to learn and classify the INSECURE targets correctly, the dataset's exaggerated bias towards the NEUTRAL target significantly hindered its ability to do so. Hence, the exact and near zero values for precision, recall, and F-score for all 4 models.

The results for the ANN classifier can be very deceiving. Although it achieved high accuracy, precision, recall, and F-score (for the NEUTRAL target) throughout all 4 models, the detection rate for INSECURE target was abysmal, not to mention the length of time that it took to train and test all 4 models. For the type and size-based models, the ANN classifier simply classified all of the instances as NEUTRAL, which explains the previously mentioned results.

The kNN classifier is a primitive algorithm and did not give any benefits above the previously explained results regarding the decision tree or the random forest classifiers. Again, the lack of beneficial results can be attributed to the state of the dataset.

The naïve bayes classifier was the worst with the Size and Full models in terms of accuracy because of the dependency assumption, but also achieved the best INSECURE F-scores for the same models. This is an important point for the purpose of this assignment. Many real-world datasets are skewed in terms of their dependent variables (targets). In most cases, the goal is to learn the properties of anomalous behavior and detect instances when they occur.

It is possible that more rows in the dataset or different models could have prevented such disappointing results. Even though none of the classifiers tested well on the supplied dataset, the best classifier was the Gaussian variant of naïve bayes. Its highest INSECURE F-scores were only 0.1450 & 0.1470 respectively, which is not acceptable. The dataset was sparse on positive sets, this contributed to the lack of successful INSECURE target detections from the models. Prudent testing with a more balanced training set would be paramount before applying to a real-world scenario.

Responsibilities Distribution:

The workload for this update was a bit uneven with Jonathan forming the core of the program. Andrew and Raunak helped as needed with adding additional functionally to the framework, verifying classification results, and writing the report.