A study on Machine Learning applied to software metrics

Ingegneria del Software 2 - Machine Learning for Software Engineering

Università degli Studi di Roma Tor Vergata

Danilo D'Amico

0320389

Index

Introduction - Aim

- The project aims to evaluate the impact of sampling techniques, cost-sensitive classifiers and feature selection techniques on the accuracy of predictive models designed to identify bugs in large-scale Apache projects.
- The projects focuses on Apache Bookkeeper and Apache Syncope

Introduction - Tools

- Fixed bugs data has been retrieved from the projects' Jira pages and GitHub commits.
- The resulting data was parsed in Java and used to create a dataset of software metrics considered useful for further elaboration in Weka
- Weka was used to evaluate the project through a Walk Forward approach and combining these possibilities for other variables:
  - Best First or no feature selection
  - Oversamplig, Undersampling, SMOTE or no sampling
  - Sensitive Threshold, Sensitive Learning or no cost sensitive classifier
  - Random Forest, Naive Bayes or IBK as classifers

Planning – Data retrieval

Closed and resolved bugs are from Jira's REST API, while GitHub commits are from a third party GitHub API available on Maven Repository: https://mvnrepository.com/artifact/org.kohsuke/github-api

Versions and Tickets are retrieved from Jira and then valid GitHub Commits are sorted into their relative version. Commits after the latest version available from Jira are ignored.

Jira Tickets and GitHub Commits are then linked together to determine Bugs.

The opening version (OV) of a Bug is identified as the latest version preceding the Ticket's creation date. Bugs with a missing or invalid OV or fixed version (FV) have been ignored.

When available and preceding the OV, I have considered the first Affected Version indicated by Jira as the injected version (IV)

Planning – Data retrieval: Statistics

- BookKeeper:
  - 419 valid Jira Tickets
  - 319 valid closed or resolved bugs
  - 13 non empty releases
- Syncope:
  - 766 valid Jira Tickets
  - 766 valid closed or resolved bugs
  - 64 non empty releases

Planning – Proportion

$$p = \frac{FV - IV}{FV - OV}$$

- When IV information wasn't available, it was computed through the Proportion method. SZZ was considered as an alternative but it is vulnerable to refactoring, missing edge cases and doesn't account for snoring.
- The project computes a Cold Start value using valid information across 5 Apache projects (Avro, OpenJPA, Storm, ZooKeeper and Tajo) and it uses it until it's ready to shift to the average value of the last 1% of bugs

pColdStart = 1,775

- Issues that are not post release (defects that have IV=FV) are excluded from the computation.
- In case FV=OV, FV-OV is set to 1 to avoid dividing by 0.

Bookkeeper has a moving window of 3

Syncope has a moving window of 8

Planning – Software metrics

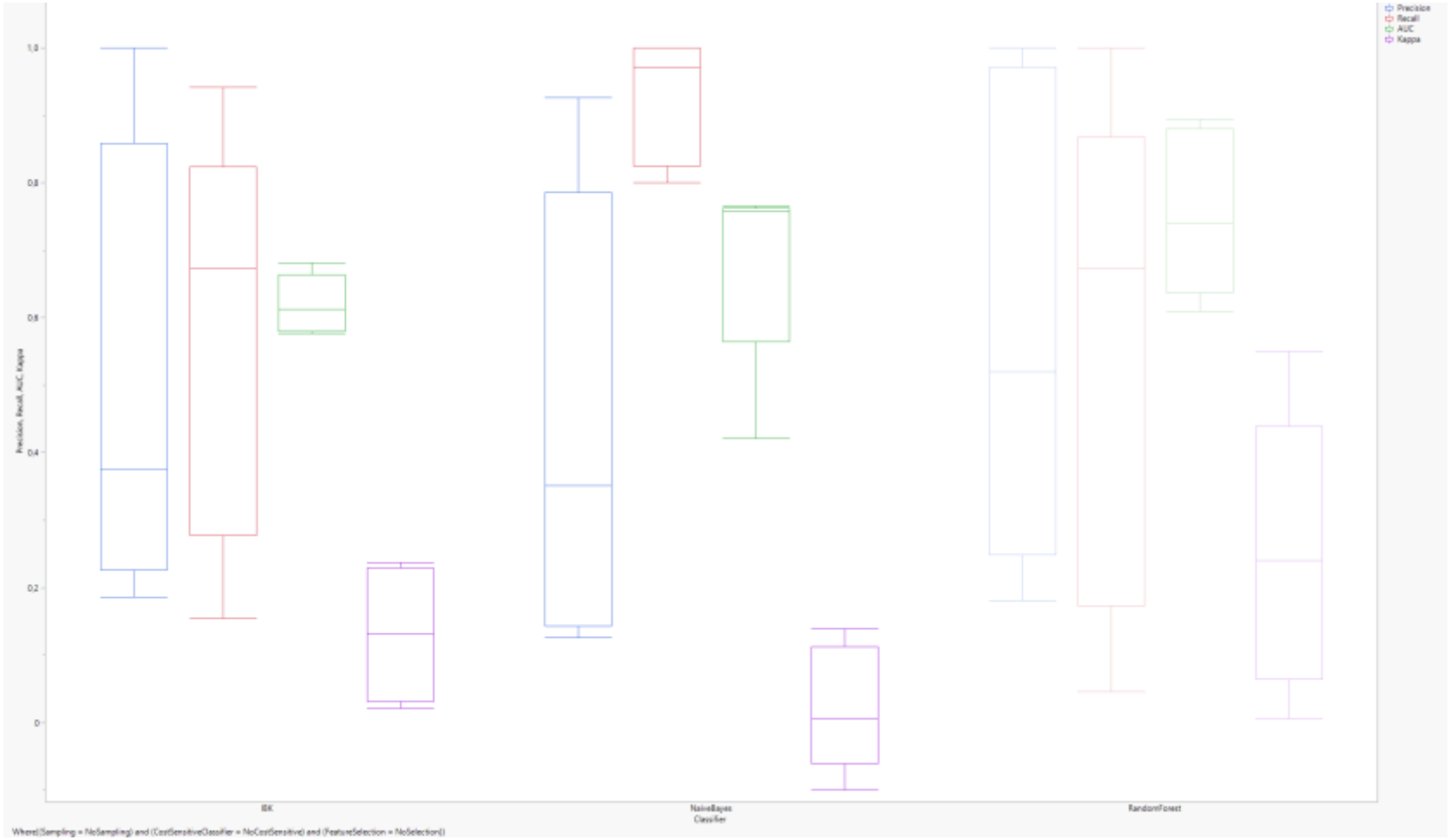
- After preparing the data, it was used to build a dataset on which to operate through Machine Learning.
- To prevent Snoring classes from lowering the precision of the training data, I have excluded the latter half of releases from the dataset itself
- Apart from the number of Authors and LOC, all class metrics have been computed from data inside the release and reset at the start of the following release, as not to have previously corrected, non-buggy classes be falsely flagged as still buggy
- The following metrics were used:

LOC, LOC touched,Number of Revisions, Number of Fixes, Number of Authors, LOC Added, Max LOC added, Average LOC added, Churn, Max churn, Average Churn, Changing Set Size, Max Changing Set, Average Changing Set

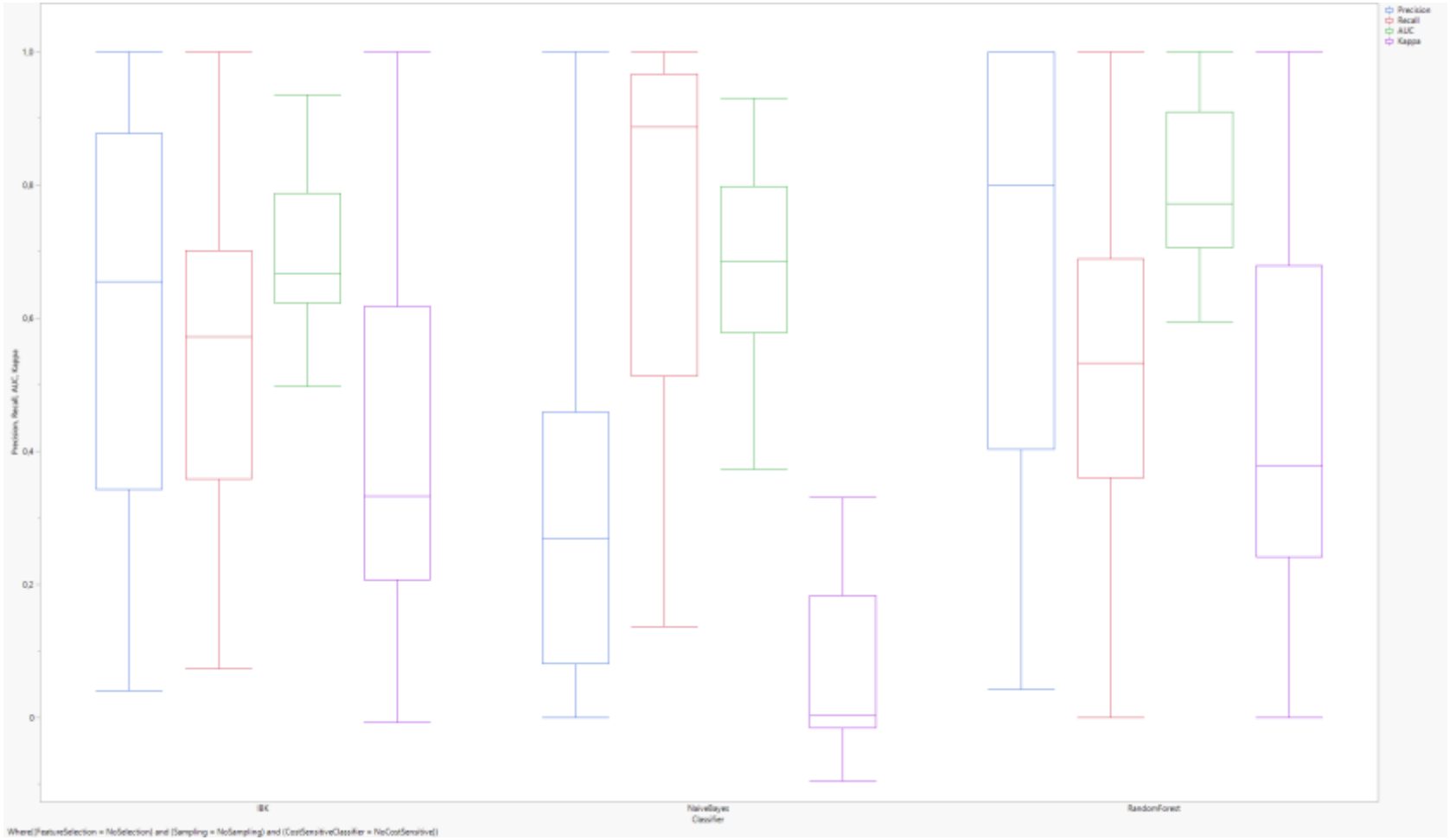
Planning - Weka

- The .arff dataset previously built is now used as input for the Weka machine learning software
- The dataset was considered as a Time Series ordered by the value «Version» and was analyzed through a Walk Forward approach. In each step, all data related to versions preceding the testing version was considered as the training set, while data related to the testing version itself was the testing set.
- Walk Forward was repeated a number of times to exhaust the combination of all Classifiers, Feature Selections, Balancing and Cost Sensitive Classifiers considered
- The following slides contain box plot graphs computed on the resulting data

Base Scenario - BookKeeper



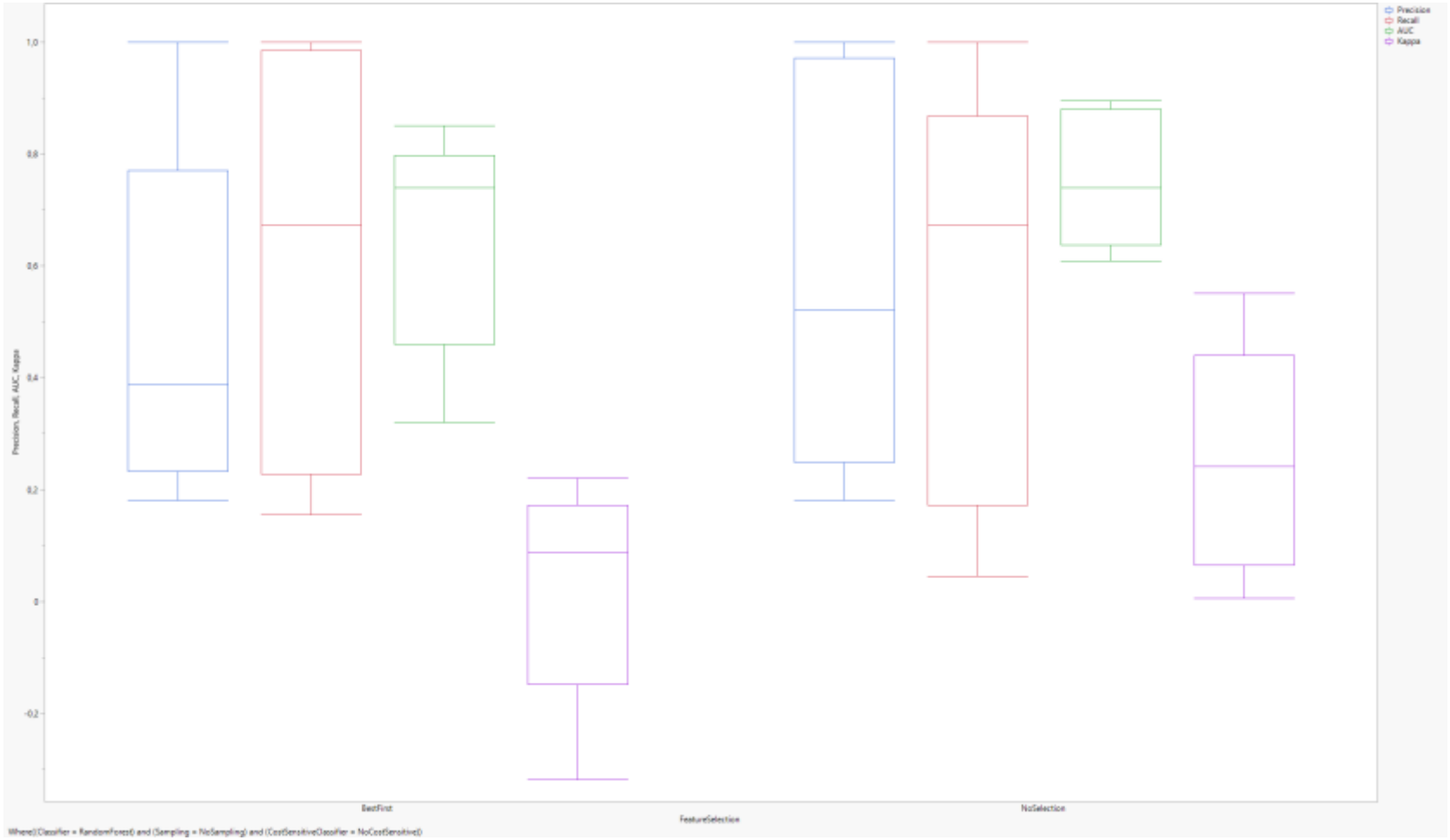
Base Scenario - Syncope



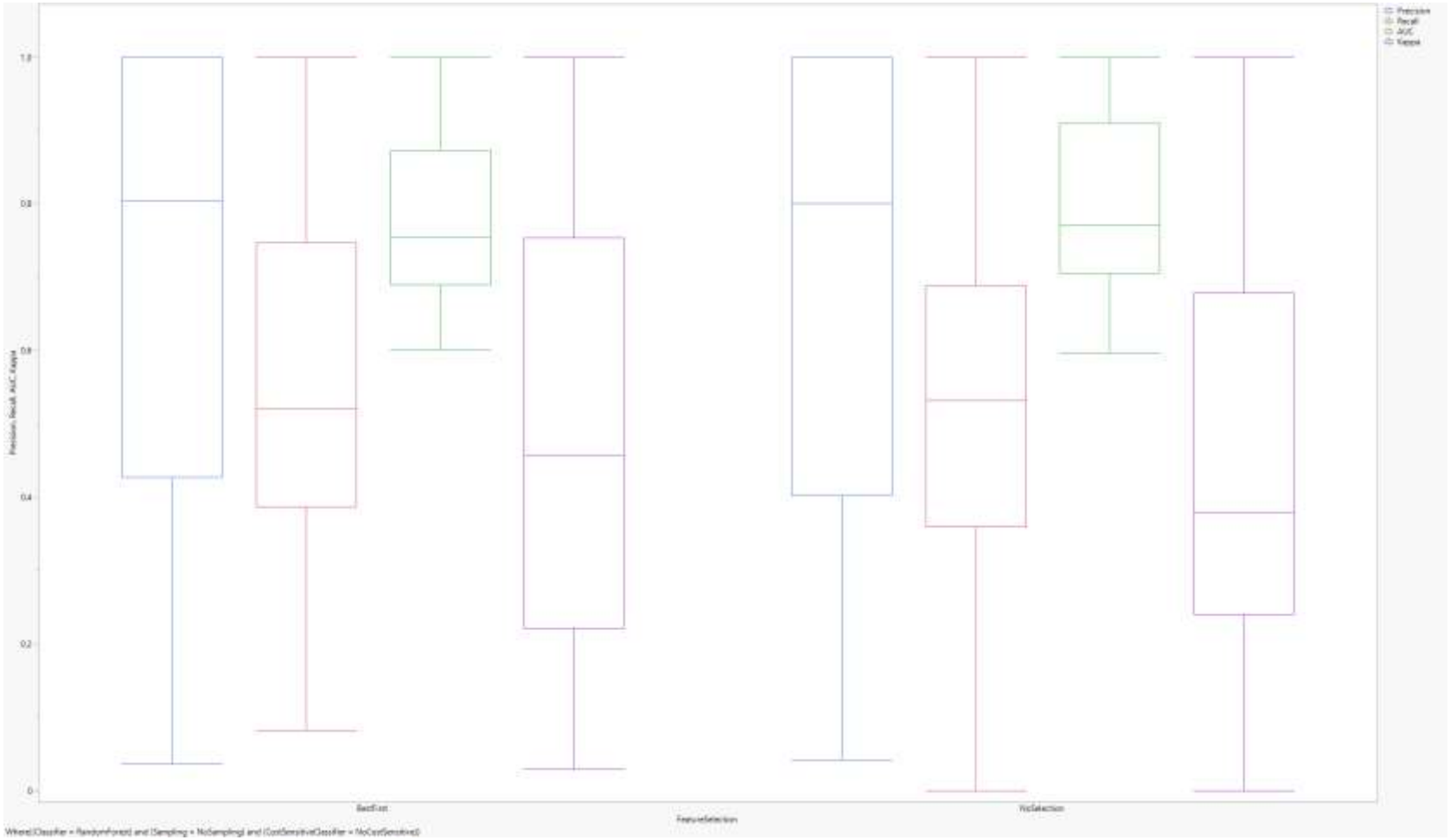
Base Scenario - Discussion

- Naive Bayes appears to have, for both projects, a Kappa that is only slightly above zero. This suggests that, unlike the other Classifiers, Naive Bayes may not be appropriate to analyze the projects as it appears nearly indistinguishable from a dummy classifier
- In BookKeeper Random Forest is a dominant classifier when compared to IBK
- In Syncope it is not possible to make the same consideration as the median IBK recall is slightly higher
- In any case, further analyses of the two datasets have been done considering Random Forest as the classifier

Feature Selection - BookKeeper



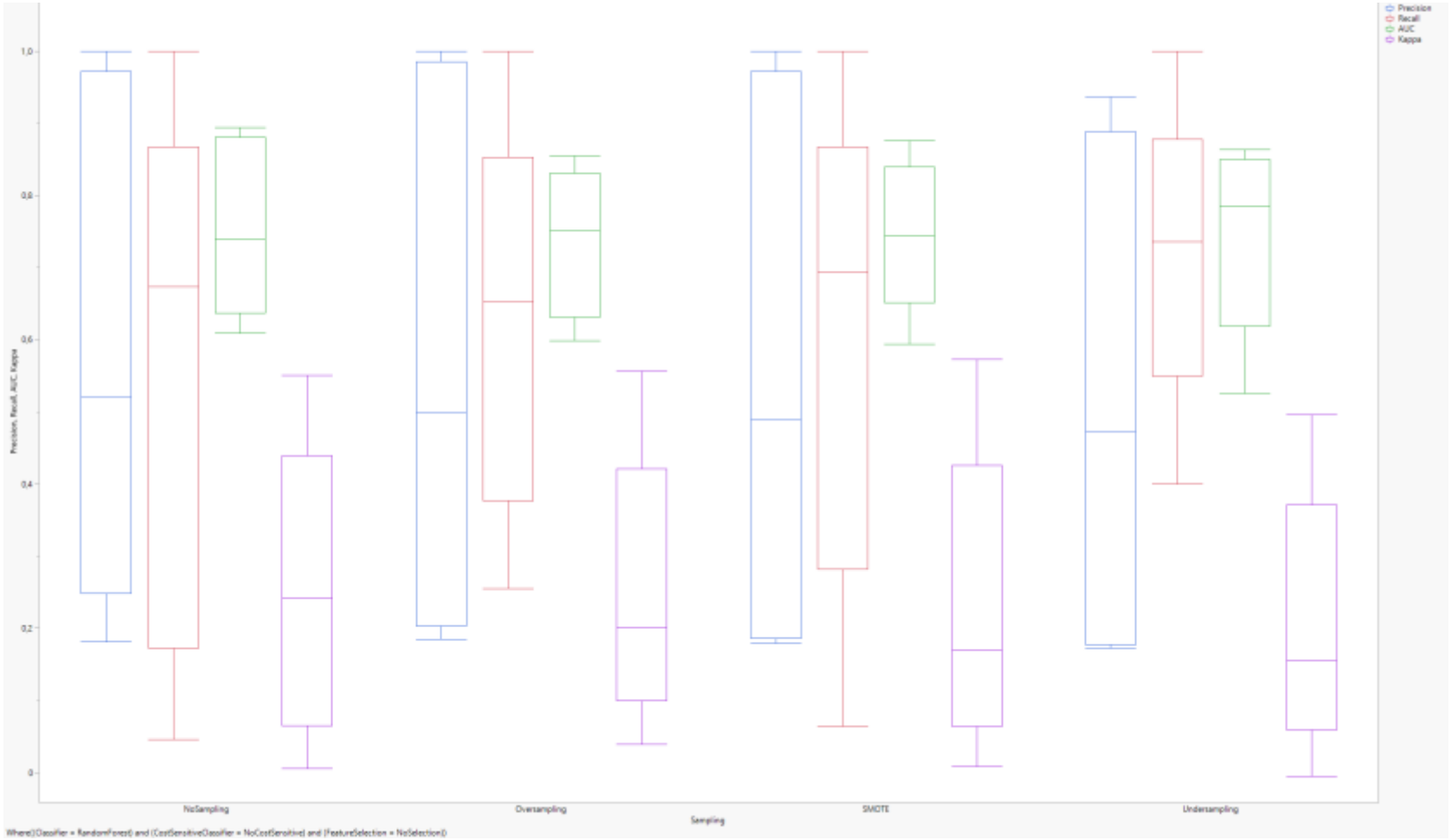
Feature Selection - Syncope



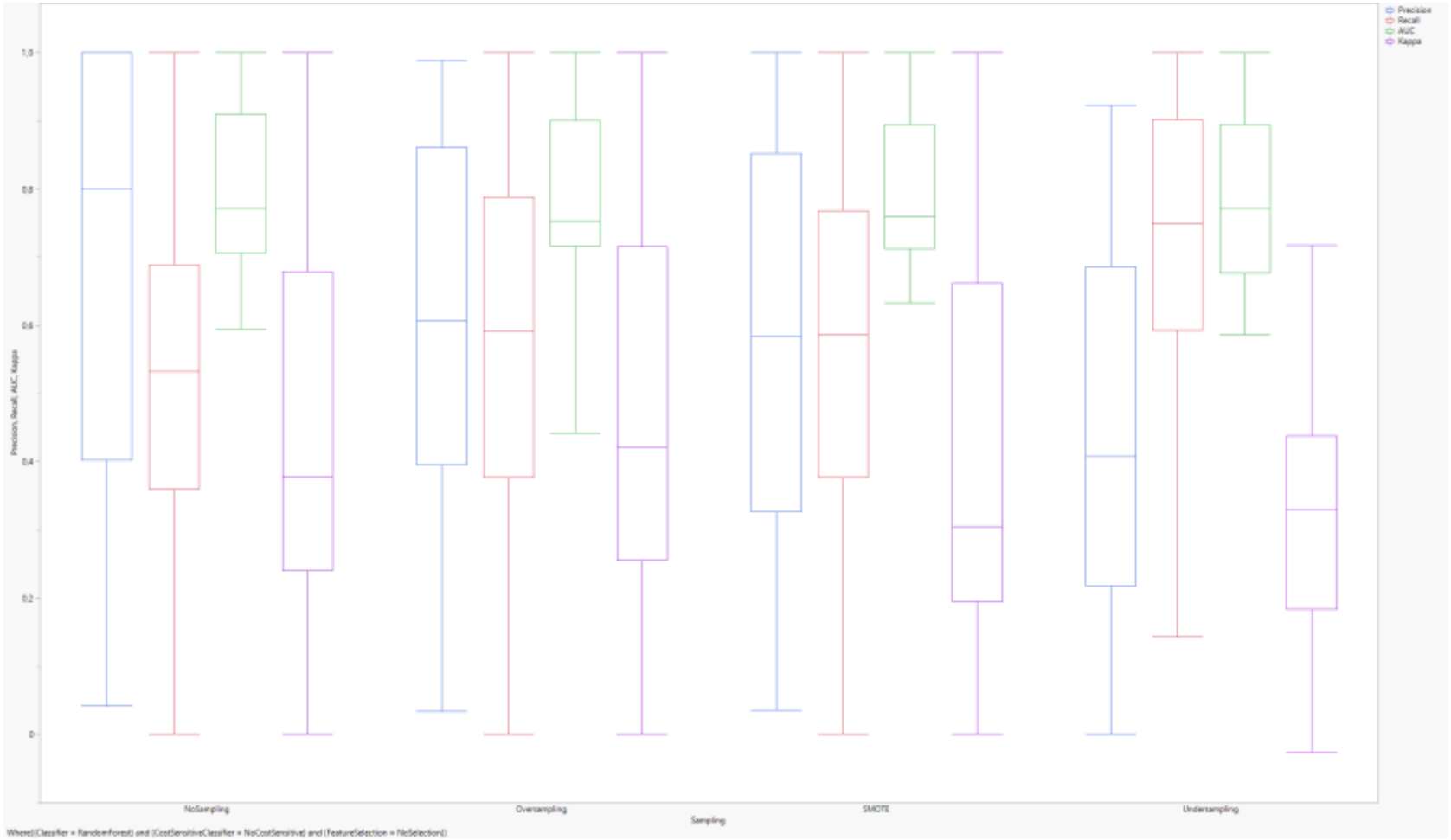
Feature Selection - Discussion

- In BookKeeper, a Best First Feature Selection lowers Kappa and Precision, meanwhile Recall remains stable and the AUC increases
- In Syncope Best First raises the Kappa instead and slightly lowers the Recall while Precision and AUC remains stable
- The difference between the two results may be in part due to the amount of data available: the lower amount of data from BookKeeper may mean removing attributes has a negative effect on the training since the most useful attributes may not have enough data to be significantly more effective at predicting the future than the remaining ones.

Balancing - Bookkeeper



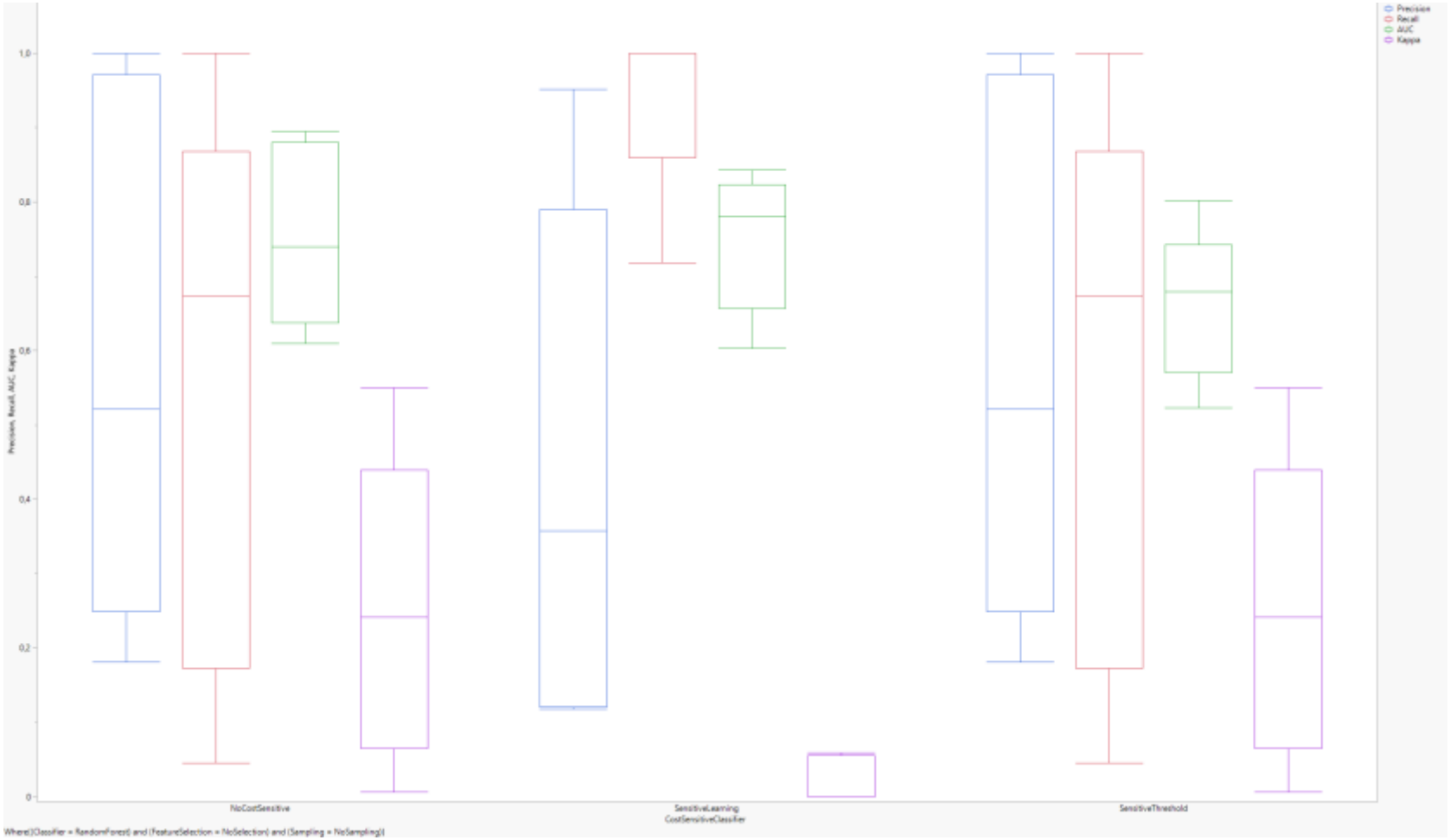
Balancing - Syncope



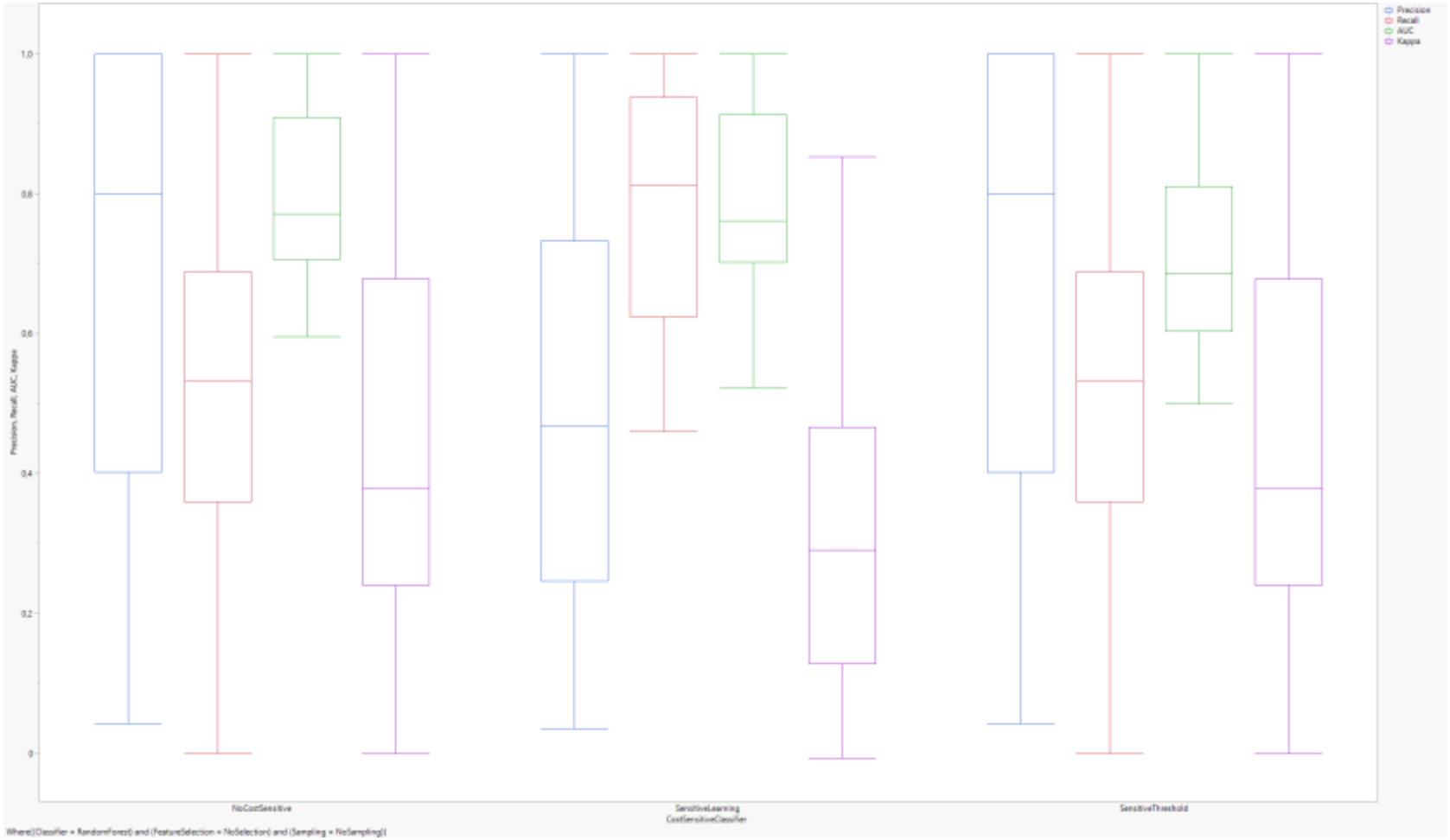
Balancing - Discussion

- Balancing has a negative effect on BookKeeper's Kappa and Precision in all considered scenarios.
  - Recall and AUC increase using Undersampling and decrease for the other sampling techniques. Undersampling is not, however, dominant because its Kappa and Precision are slightly lower than Oversampling's and SMOTE's.
- In Syncope we can see a similar effect in both Kappa and Precision.The exception is Oversampling: the Kappa is actually higher than the base scenario, an effect that can be attributed to a higher availability of minority classes compared to BookKeeper
  - AUC is lower while the Recall is higher, especially in Undersampling
- The effect on Kappa and Precision is expected, since changing the ratio between minority and majority classes in the training set inevitably changes the model's expectations on the testing set buggyness

Cost Sensitive Classifier - Bookkeeper



Cost Sensitive Classifier - Syncope



Cost Sensitive Classifiers - Discussion

- Sensitive Learning, in which the Cost Matrix has a False Negative weight of 10 times the False Positives, shows the goal of Cost Sensitive Classifiers: increasing the Recall value at the expenses of everything else. AUC is the only value that slightly increases in BookKeeper

- Increasing the weight of False Negatives in Sensitive Threshold we can see the Recall increase while the Precision lowers, indicating that the model increasingly identifies more and more classes as «buggy» in an attempt of avoiding the penalty hit from False Negatives

Key Takeaways

BookKeeper	Feature Selection	Balancing	Cost Sensitive Classifier
Precision	No Feature Selection	No Balancing	No Cost Sensitive Classifier
Recall	Best First	Undersampling	Sensitive Learning
AUC	Best First	Undersampling	Sensitive Learning
Kappa	No Feature Selection	No Balancing	No Cost Sensitive Classifier

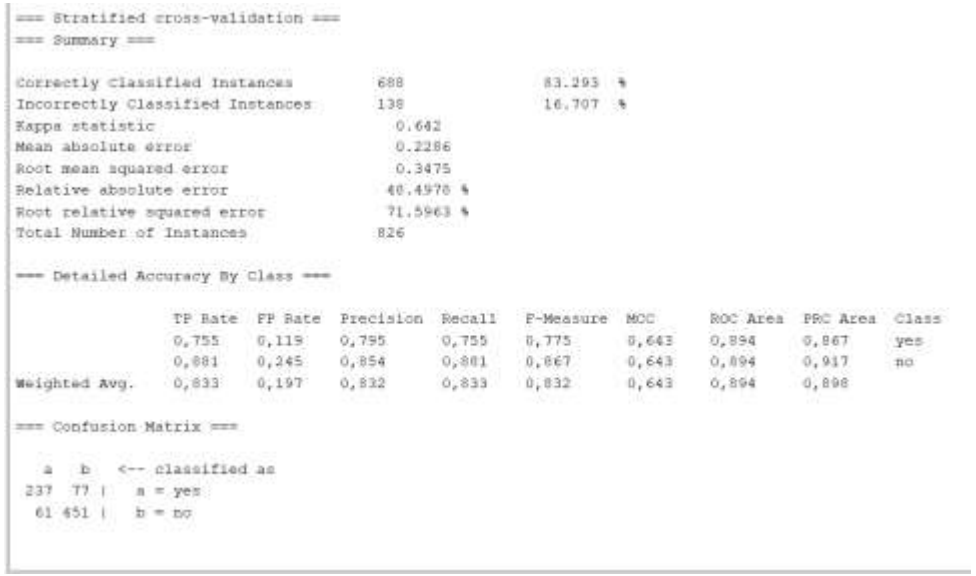
Syncope	Feature Selection	Balancing	Cost Sensitive Classifier
Precision	Best First	No Balancing	No Cost Sensitive Classifier
Recall	No Feature Selection	Undersampling	Sensitive Learning
AUC	No Feature Selection	No Balancing	No Cost Sensitive Classifier
Kappa	Best First	Oversampling	No Cost Sensitive Classifier

Threats to validity

- It is unknown of much of the difference in results between BookKeeper and Syncope is due to the different amount of data the two projects have available
- Data between Jira and GitHub is sometimes conflicting, forcing a choice between a reduction in the size of the dataset or on the fly adjustments to potentially corrupt or inaccurate data
- Different metrics may have generated a dataset where the relationship between classifiers is reversed, meaning that the apparent dominance of Random Forest may be highly dependant on arbitrary choices

Threats to validity – Not a Time Series?

- An analysis through Weka UI of the BookKeeper dataset returns an impressively high Kappa when compared to Walk Forward.
- To run Random Forest with 10-folds cross validation the File Name value was removed as string values are not compatible with the classifier, meaning that the metrics analyzed are independent from the history of the files themselves
- This suggests that training a model on a number of available projects and running it on another could return useful results



```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         688               83.293 %
Incorrectly Classified Instances       138               16.707 %
Kappa statistic                          0.642
Mean absolute error                      0.2286
Root mean squared error                  0.3475
Relative absolute error                 46.4978 %
Root relative squared error             71.5963 %
Total Number of Instances              826

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0,755     0,119     0,795       0,755   0,775      0,643    0,894     0,867     yes
               0,881     0,245     0,854       0,881   0,867      0,643    0,894     0,917     no
Weighted Avg.  0,833     0,197     0,832       0,833   0,832      0,643    0,894     0,898

=== Confusion Matrix ===

   a   b   <-- classified as
 237  77 |   a = yes
  61 451 |   b = no
```

Links

GitHub repository: https://github.com/DaniloDamico/ISW2Milestone

SonarCloud report:
https://sonarcloud.io/project/overview?id=DaniloDamico_ISW2Milestone