UNIVERSITY OF PASSAU
Chair of AI Engineering
https://www.fim.uni-passau.de/ai-engineering/

Mining Software Repositories Lab
WS 2022/2023
Prof. Dr. Steffen Herbold, Dr. Alexander Trautsch, Lukas Schulte

Exercise 3 · Due at 2023-03-17

General information

The task groups for specific days are a rough estimate. If you are finished with tasks for Monday you should start on the tasks for Tuesday. The faster you are working through the extraction and data analysis the faster you can explore more data and start on the presentation and maybe even prepare information for the final report.

Hint: The tool used in this exercise is SLOW. You should start as early as possible with the extraction! You can also use the computers in this room instead of a laptop.

Be aware that the tasks described for each weekday are only a **minimal** solution. You have a lot of time in this course and should explore more options. Some suggestions: Does readability correlate with complexity? How does readability change when fixing a bug? You can also include the plugin from the first week and see whether readability increases if SATD is removed. These are just some questions that you can explore and mention in the presentation of the results and in the final report.

Tasks

Research questions this week:

RQ1 Do developers increase readability deliberately?
Investigate changes in readability for a file and look at the commit message.

RQ2 Can we predict whether the readability of a file will be improved?
Build a predictive model taking features from the commit message to predict whether the readability will improve.

Monday

1. Create a private Gitlab Repository[1] for the week and invite the lecturer.
2. Download the readability tool from the paper presented in the introduction[2].
3. Create a python environment[3] and install Pydriller[4].
4. Create a python script that can be executed with a repository URL as an argument, e.g., main.py –repo https://github.com/apache/commons-vfs

[1] https://git.fim.uni-passau.de
[2] https://dibt.unimol.it/report/readability/
[3] https://www.freecodecamp.org/news/how-to-setup-virtual-environments-in-python/
[4] https://pypi.org/project/PyDriller/

5. The python script should then use Pydriller to load the repository and iterate over the commits.
   Required: Only extract the master branch and only modifications in .java files via Pydriller.
   Required: Only extract commits up to and including the committer date of 2022-12-31.

6. Extend the python script so that it extracts more information via Pydriller, extract at least: commit message, commit hash, author name, author email, committer date, number of changed files for each commit. Moreover, extract for each file: path, complexity, nloc (all via Pydriller). Remeber: We are only interested in files that were modified (not added, deleted, renamed, etc.).

Tuesday

1. Extend the python script so that it can extract the readability for each changed file, extract the readability before and after the change. This means that you need to temporarily save the before and after version of the contents of the files provided by Pydriller and then run the readability tool with the list of files.
   Required: The readability tool is slow as it requires to start a Java VM. You need to give it a list of files that contain the before and after contents for all changed files! Otherwise the mining will not finish in time!

2. Test this for a smaller Apache repository[5].
   Hint: Try to extract everything for one commit with 2 files first before you run this for the whole repository.

3. Extend the python script to save the data you gather to one file per repository. Complete the mining process for commons-vfs[6], commons-bcel[7] and commons-codec[8].

4. While this runs (expect hours of run time for each repository!) create a new python environment in a new folder notebooks. Install Jupyterlab, Pandas, Scipy, Matplotlib and Scikit-learn into the new environment.

Wednesday

1. Create a Jupyter notebook that can read the data from the files that you saved in the mining process.

2. Create two Wordclouds from the commit messages, one contains only the commits where readability has improved and the other one only commits where readability decreased.
   Hint: You need to perform text preprocessing steps, e.g., text cleaning, stop word removal, and potentially stemming.

3. Think about what you would expect a developer to write in the commit message if he or she wants to improve the readability. Search through your available data for this and

[5] https://github.com/apache/commons-vfs
[6] https://github.com/apache/commons-vfs
[7] https://github.com/apache/commons-bcel
[8] https://github.com/apache/commons-codec

find whether the readability was actually improved in the changes (if you find intentional readability improvement). You can use the Wordclouds to give you hints.

4. Create a set of keywords which, in your opinion, are able to signify the intent of the developer to increase readability. Use these keywords to identify commits where the intent of the developer is to increase readability for RQ1.

5. Create a boxplot of both populations and compare whether their readability was actually changed and how large the effect is.
   Hint: Check sample distribution and apply an appropriate statistical test, you can refer to[9].

Thursday

1. Create a second Jupyter notebook (or extend your first one) that reads the data and create a Random Forest model that uses the commit message to predict whether the readability delta is negative or not, i.e., if readability is improved or not.

2. Look at the feature importances of the created model. Look whether you can find keywords that are indicative on whether the model predicts readability improvement that you have missed.

3. Evaluate the model performance in a cross-project setting for all projects, e.g., evaluate for commons-vfs with only data from commons-bcel and commons-codec.

4. Evaluate the model performance in a within-project setting for all projects, e.g., evaluate commons-vfs change for a file with only the data that predates the change.
   Hint: This means you can only include data before the currently predicted change.

5. Use the model performance evaluation to answer RQ2, i.e., is the model able to predict readability increase.

6. Finish up extraction and evaluation tasks and make sure you have updated the code on your Gitlab for this project with the current version.

7. Make sure that you can answer the research questions using your findings from this week.

8. Prepare a presentation with your results. Make sure you include a sound motivation, describe your methodology, data analysis and results as well as implications.

Friday

1. Finish up last changes.

2. Send your presentation to the lecturer prior to your presentation slot.

3. Present your results.

[9] https://sherbold.github.io/intro-to-data-science/11_Statistics.html