Topic: developer ranking

Team Number: 02

Class: 1

Members and Student ID:

Zhong Xin: 320180940671

Li Peidun: 320180939901

Lin Xiaoduo: 320180940020

Liu Lingxiao: 320180940040

Hypothesis and the mapping to data

Assumptions: Within a certain period of time, it is assumed that the same developer's number of file modifications (modules) and the bug rate are inversely proportional to the developer's quality

1. Requirements and design

Requirements:
1. Crawl data from Linux-Kernel
2. Clean and filter the data
3. Select an appropriate prediction model
4. Establish prediction model and output chart
5. Compare the predicted value and the true value
6. Evaluate the results

Design:
1. Crawl Developer data on Linux-Kernel
2. Cleans up data that is not related to the prediction model, retaining developer names and submission information
3. The linear regression model is selected as the regression model
4. Implement the data output with python specific packages
5. The correlation coefficients are calculated to determine the degree of fit
6. Evaluate model results and explain the correlation of data

2. Data harvesting concept and implementation

Data harvesting is a process of using small scripts to automatically extract large amounts of data from certain websites and use it for other purposes. Data harvesting or network scraping has become a concern for website operators and data publishers. When capturing some data, you can speed up subsequent captures by capturing only the changed or newly generated data objects.
Incremental harvest new, modified, and deleted data objects on the index volume or file server. Because the harvest is incremental, it takes less time to update the metadata repository, and the load on the system is lighter than the original harvest. In our group project, we crawled the names of the developers in Linux-Kernel and the number of submissions. After obtaining these data, we could proceed to the next step.

Git has many commands to crawl data, so we tried the following command and finally decided what to crawl
%H the complete hash of the commit object (COMMIT)
%h commits the short hash string of the object
The full hash string for the %T tree object (tree)
A short hash string for the %t tree object
The full hash string of %P parent

A short hash string for the %p parent object
% The name of an author
% ae author's email address
% ad author revision date (can be customized with the -Date = option)
% ar author revision date, as shown how long ago
% cn Name of the committer
% ce submitter's email address
% cd submission date
% cr submission date, as shown how long ago
%s Submit instructions
We used $git log > commit.txt to get the developer's commit record.
The git log - pretty = format: "% an" & gt; date.txt is used to get information about the submitted author.
Git log --format='%aN' | sort -u | while read name;Do the echo - en $name "\ t";
Git log --author="$name" --pretty=tformat: --numstat | awk '{add += $1;Subs + = $2;Removed lines: %s, total lines: %s\n", add, subs, LOc}';Done is used to crawl the author's specific modifications, such as the number of lines of code added, the number of lines of code removed,and output these results.

3. Data cleaning concept (analysis and ”fixing”) and implementation

Data cleaning is the process of preparing data for analysis by removing or modifying data that is incorrect, incomplete, irrelevant, duplicated, or improperly formatted. This data is usually not necessary or helpful when it comes to analyzing data because it may hinder the process or provide inaccurate results.

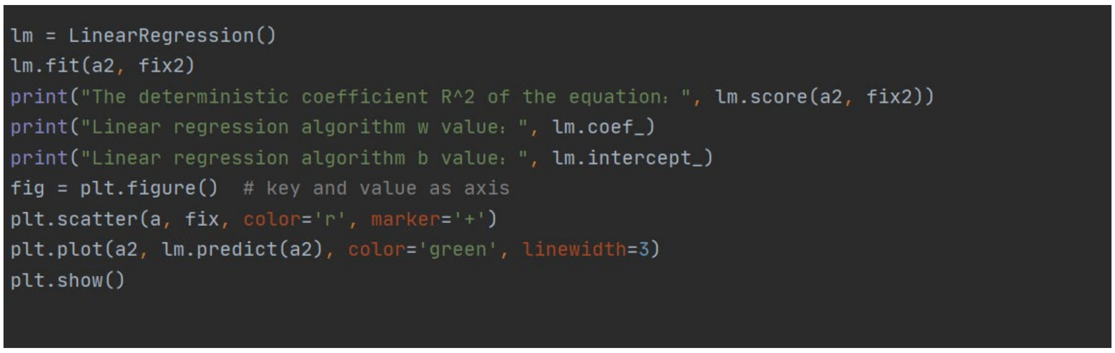
Here are several key benefits that come out of the data cleaning process:
It removes major errors and inconsistencies that are inevitable when multiple sources of data are getting pulled into one dataset.
Using tools to cleanup data will make everyone more efficient since they’ll be able to quickly get what they need from the data.
Fewer errors means happier customers and fewer frustrated employees.
The ability to map the different functions and what your data is intended to do and where it is coming from your data.

Here are some of our specific steps for data cleaning:
1. Clarify errors: Clean up outliers and filter out data that is too far from the model
2. Standardize the whole process: In the process of data modeling, try to standardize, so that the established regression model and regression relationship can better fit the real value data, make the prediction more convincing and credible.
3. Validate Accuracy: After cleaning some outliers, the regression model is output repeatedly to verify the validity of data cleaning.
By comparing the models before and after the cleaning, the paper analyzes whether the cleaning has played a role in improving the degree of regression fitting.

4. Clean up unwanted data: Delete the data that is not necessary for the regression model, and keep the content that is useful. After this stage, we can obtain a copy of the data that is only relevant to the establishment of the regression model.
5. Analyze the data: After cleaning up, only the data set of the author and the modified record is left, and the regression model will be built for these two data.
6. Communicate with the Team: We need to talk to our team members and ask them if they have any other ideas about how to clean up the data that they've got. In the course of a team project, there may be different opinions on the data currently obtained, some people may think that the current data is not of practical significance to establish a regression model, or the current different data is not very relevant, so the communication between the teams is very important.

4. Technology selection and technology evaluation

1. We chose to build a linear regression model to predict whether the two selected variables are related to each other.



```
lm = LinearRegression()
lm.fit(a2, fix2)
print("The deterministic coefficient R^2 of the equation: ", lm.score(a2, fix2))
print("Linear regression algorithm w value: ", lm.coef_)
print("Linear regression algorithm b value: ", lm.intercept_)
fig = plt.figure()  # key and value as axis
plt.scatter(a, fix, color='r', marker='+')
plt.plot(a2, lm.predict(a2), color='green', linewidth=3)
plt.show()
```

2. The results obtained do not match our expectations, so we cannot choose a linear regression model to predict

5. Implementation

Coding style: python

6. Interpretation of results

The result does not meet our prediction: within a certain period of time, the number of file modifications (modules) and error rate of the same developer is inversely proportional to the quality of the developer. We can’t get a developer ranking from this.

7. Problems during the project

1.When we tried to crawl data from Linux-Kernel, we did not know what type of data could be used by us, so we inquired related materials and communicated our ideas with classmates, so as to roughly understand the different data types in Linux-Kernel.
2.After we understand the data type of Linux-Kernel, we start to consider what type of data to crawl, so that we can build a prediction model.
The model can not only cover some elements of the project development, but also

follow the normal logic.
3.After consideration, we intend to build a predictive model for Linux-Kernel developer quality. Developers are known to come and go in terms of their individual abilities, and their contributions to Linux-Kernel code vary. Therefore, we need to find some criteria to judge.
4.The developers of Linux-Kernel will submit code in the development process, and their code will also have many bugs, and the number of bugs reflects the quality of developers to some extent, so we plan to crawl the bug data of developers' code.
5.In the process of crawling bug data, we found that similar data did not directly appear in Linux-Kernel. After careful searching, we found that the word "fix" appeared repeatedly in the personal records of many developers. The bug must have been fixed, so the amount of fix data also reflects the quality of the developer.
6.After we crawled the data, the scale of the data shocked us. We have reached a TXT file of more than 500 MB, the data volume is very large. Therefore, our data cleaning work is bound to be very heavy.
7. After removing the data that was useless for building regression models, we got a data set containing developers and their submissions. At this point, we began to consider what regression model to use to fit the correlation between the two. After trying different models, we decided to use linear regression.