SPL-1 Project Report, 2019

Programmer Deanonymization Using Code Stylometry

SE 305 : Software Project Lab I

Submitted by
Yasin Sazid
BSSE Roll No. : **1006**
BSSE Session: 2017-2018

Supervised by
Dr. Kazi Muheymin-Us-Sakib
Designation: Professor
Institute of Information Technology

Institute of Information Technology
University of Dhaka
29-05-2019

Table of Contents

1. Introduction

This project is about attributing authorship to source codes, that is, deanonymizing programmers based on their coding styles. This can be helpful in software forensics, detecting plagiarisms, resolving copyright conflicts etc. Each programmer learns to code in a unique way that results in the coding style of distinguishing "fingerprints”, these fingerprints can be used to compare known programmers' codes with an anonymous piece of source code to find out the anonymous code was authored by which one of the known programmers. I tried to deanonymize programmers of first year BSSE students using the above idea.

This is a primary application of machine learning where I used machine learning algorithms like Naïve-Bayes and K Means clustering algorithm to make a prediction about the authorship of a source code based on layout features which are extracted directly from the source code.

1.1. Background Study

To implement this project, some prior study was necessary –

Programmer Deanonymization

I studied a research paper named “De-anonymizing Programmers via Code Stylometry” where researchers made a break-through in deanonymizing source code authors by extracting syntactic, layout and lexical features from source codes and using machine learning techniques like random forest. U.S. Army Research Laboratory was involved in this research and this technology has great impacts on software forensics, authorship attribution and copyright infringement cases. I studied their approach and followed it in smaller scale. I used K Means and Naïve-Bayes (Beta distribution, normal distribution) to implement the deanonymization part.

Beta Distribution

Some features of the feature set of this project is presented with percentage points. For Naïve-Bayes classifier, those features had to be assumed to be following beta distribution. So I studied how beta distribution, beta function, parameters of these distribution, their relations with mean and variance etc. work and how to calculate these things. I had to use online beta distribution calculators to simulate these calculations for my dataset to check whether I calculated them correctly in my project or not.

Normal distribution

Some features has been assumed to be following normal distribution for Naïve-Bayes classifier. I studied how to calculate normal distribution value for a data point and its relation with probability.

Naïve-Bayes classifier

It is a simple classifier based on the distributional property of data. Every feature is thought to have same weight in this procedure. It gives a relative probability for each class of data points, given an unknown data point. I studied on how to use it, formulas, assumptions and how to implement it in my project efficiently.

K Means algorithm

K Means algorithm is used to find clusters within a given set of data points. I had to study K Means and different versions of it thoroughly and decide on how to integrate it in my project efficiently as it was an important part of my project. Standard K Means algorithm is simple to understand, but for my project, data points were much more complex and it was difficult to manage data points from programmer profiles to clusters of source file feature sets. I used this algorithm to clustering and then created ranking based on the cluster containing the data point for anonymous source code.

1.2. Challenges

There are a number of challenges in implementing a new software solution. The process can be confusing and overwhelming. There are many challenges I have faced to implement this project. There are some of them –

- ➢ Working with header files for the first time
- ➢ Working with multiple source files
- ➢ Statistical learning –
    - ▪ Distributional properties –
        - o Beta distribution: when and how to use
        - o Normal distribution
    - ▪ Machine learning algorithms –
        - o Naïve-Bayes classifier: how to use
        - o K Means clustering algorithm: how to use and make a ranking based on clustering
- ➢ Handling delicate data, i.e., very large or small numbers for statistical purposes
- ➢ Handling complex data points for K Means clustering
- ➢ Hundreds of sensitive calculations and conversions with internal dependencies made it difficult to find errors

- Parsing data from CSV file
- It was a challenge to decide on how to process input source codes and how the approach should be to extract different features

2. Project Overview

There are basically two parts in my project –

- **Extracting features** from source codes
- **Deanonymization** using machine learning classifiers

2.1. Extracting features

Feature set for this project focused on layout features such as –

- Bracing style
- Percentage of proper indentation
- Rate of white space in function or loop signatures
- Space after/before certain symbol (equal '=' symbol)
- Total white space percentage
- Total empty lines percentage

To extract these features from a source code, different feature analyzers have been written which take lines of codes as input and outputs the quantitative or qualitative feature. While making a profile for a programmer, source codes written by that programmer are analyzed by these analyzers and code-wise features are stored as record corresponding to that programmer's identification number. This is the training data for the deanonymization part. A profile can be updated by more training data even after the creation.

2.2. Deanonymization

This is the part of the project where programmer profiles are matched with the style of an anonymous source code in order to deanonymize the author of that anonymous code. For this, two different paths have been followed in my project –

- Naïve-Bayes classifier
- K Means clustering algorithm

2.2.1. Naïve-Bayes classifier

Naïve-Bayes classifier is a "probabilistic classifier" based on the application of Bayes' theorem with strong (naive) assumptions of independence between features. This classifier treats every feature equally. To implement it, firstly, distribution of data has to be studied. Some of the feature set had features in percentage points. For this kind of features, beta distribution has been assumed and normal distribution has been assumed for other features. Then based on those distributions, features of the anonymous source code is matched with the profiles of programmers to get a ranking sum indicating the relative probability of each programmer writing that source code.

2.2.2. K Means clustering algorithm

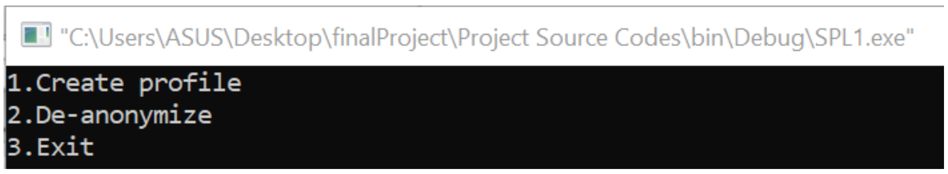
K Means algorithm has been implemented to make k = (number of programmer profiles + 1 (for the anonymous source code)) clusters within data points indicating the feature set of each training data file. Then the cluster consisting the anonymous code is taken and identification numbers of data points are extracted. From those identification numbers, we can conclude how many data points a certain programmer owned in that cluster. Now the programmer with higher (number of owned data points within that cluster/total number of owned training data points) ratio has higher probability to own the anonymous source code.

3. User Manual

Users can **"create a profile"** for a programmer by providing training data (source code files) or they can **"deanonymize"** an anonymous source code with the present list of programmers by providing the anonymous source code file.

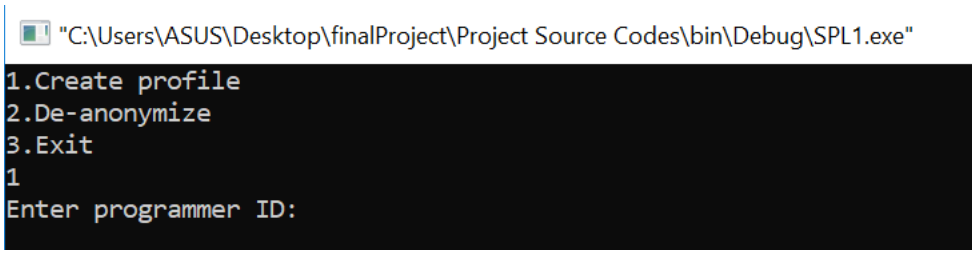
3.1. Creating a profile

- Open the executable (.exe) file
- You will see an interface



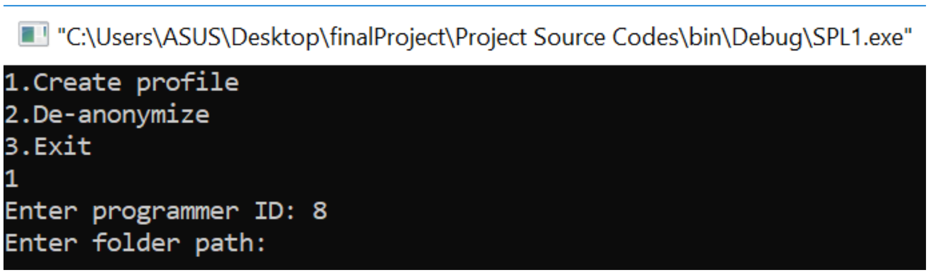
```
"C:\Users\ASUS\Desktop\finalProject\Project Source Codes\bin\Debug\SPL1.exe"
1.Create profile
2.De-anonymize
3.Exit
```

- Enter "1"



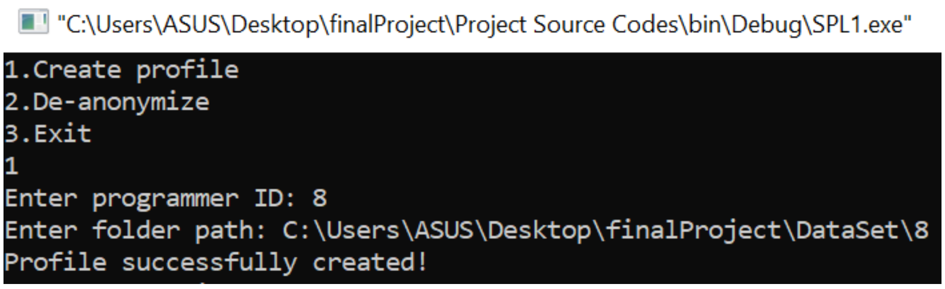
```
"C:\Users\ASUS\Desktop\finalProject\Project Source Codes\bin\Debug\SPL1.exe"
1.Create profile
2.De-anonymize
3.Exit
1
Enter programmer ID:
```

- ➢ Enter a unique programmer ID



```
"C:\Users\ASUS\Desktop\finalProject\Project Source Codes\bin\Debug\SPL1.exe"
1.Create profile
2.De-anonymize
3.Exit
1
Enter programmer ID: 8
Enter folder path:
```

- ➢ Enter the folder path for the training data (source code files)

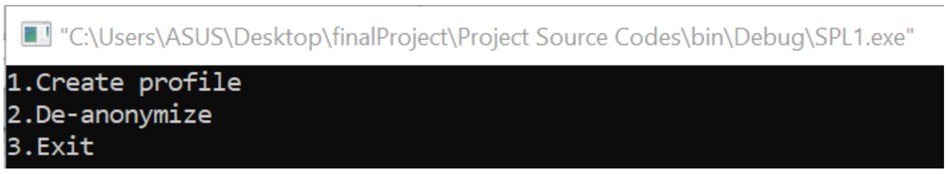


```
"C:\Users\ASUS\Desktop\finalProject\Project Source Codes\bin\Debug\SPL1.exe"
1.Create profile
2.De-anonymize
3.Exit
1
Enter programmer ID: 8
Enter folder path: C:\Users\ASUS\Desktop\finalProject\DataSet\8
Profile successfully created!
```

- ➢ A profile with the given ID and data have been created
- ➢ Enter “3” to exit

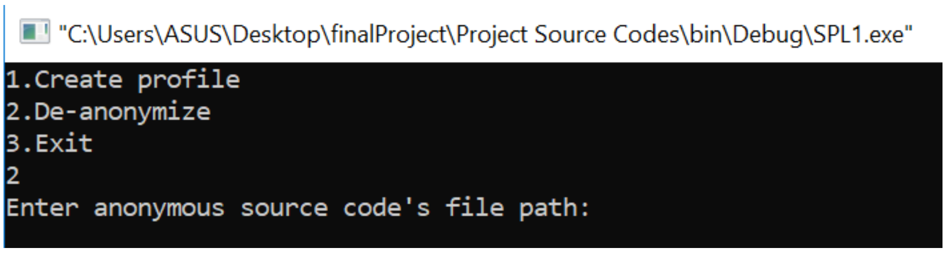
3.2. Deanonymizing

- ➢ Open the executable (.exe) file
- ➢ You will see an interface



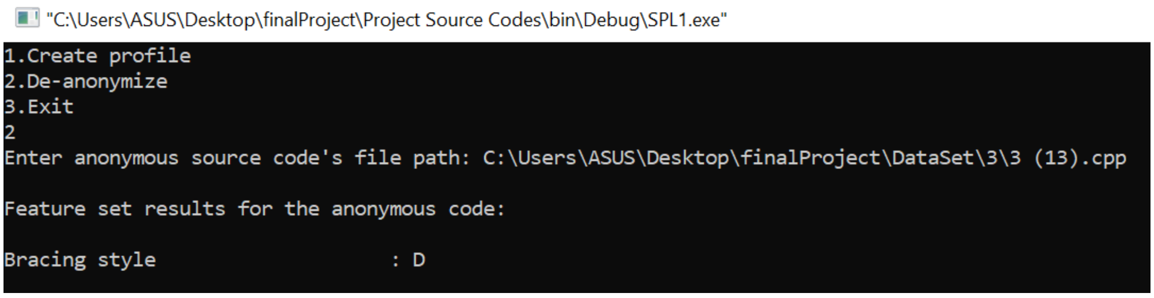
```
"C:\Users\ASUS\Desktop\finalProject\Project Source Codes\bin\Debug\SPL1.exe"
1.Create profile
2.De-anonymize
3.Exit
```

- ➢ Enter “2”



```
"C:\Users\ASUS\Desktop\finalProject\Project Source Codes\bin\Debug\SPL1.exe"
1.Create profile
2.De-anonymize
3.Exit
2
Enter anonymous source code's file path:
```

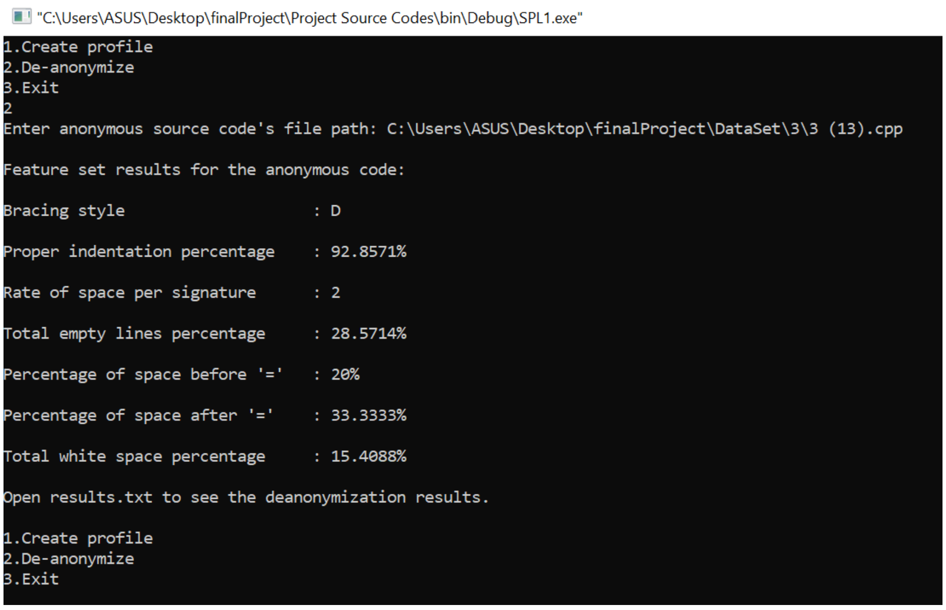
- Enter the path of the file you want to deanonymize



"C:\Users\ASUS\Desktop\finalProject\Project Source Codes\bin\Debug\SPL1.exe"

```
1.Create profile
2.De-anonymize
3.Exit
2
Enter anonymous source code's file path: C:\Users\ASUS\Desktop\finalProject\DataSet\3\3 (13).cpp

Feature set results for the anonymous code:

Bracing style                  : D
```

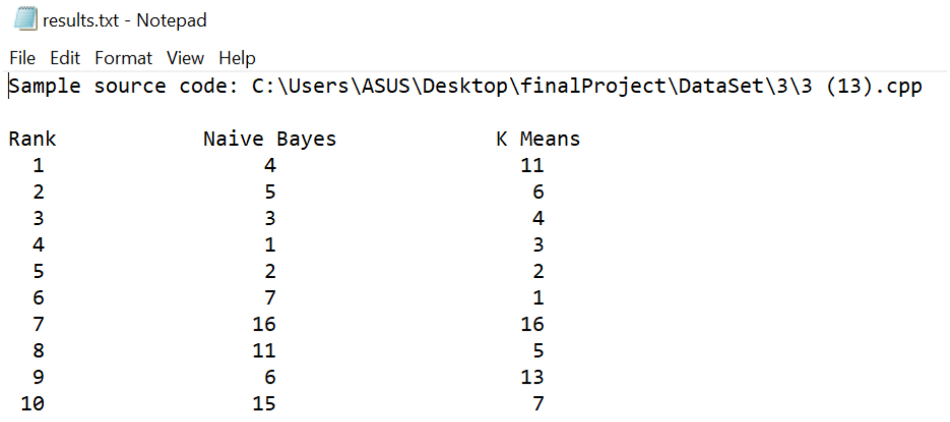
- You will see feature set results for the anonymous code.
- Continue pressing “Enter” for each feature statistic.



"C:\Users\ASUS\Desktop\finalProject\Project Source Codes\bin\Debug\SPL1.exe"

```
1.Create profile
2.De-anonymize
3.Exit
2
Enter anonymous source code's file path: C:\Users\ASUS\Desktop\finalProject\DataSet\3\3 (13).cpp

Feature set results for the anonymous code:

Bracing style                  : D

Proper indentation percentage    : 92.8571%

Rate of space per signature      : 2

Total empty lines percentage     : 28.5714%

Percentage of space before '='   : 20%

Percentage of space after '='    : 33.3333%

Total white space percentage     : 15.4088%

Open results.txt to see the deanonymization results.

1.Create profile
2.De-anonymize
3.Exit
```

- Enter “3” to exit
- Open the file named “results.txt” for the deanonymization results



results.txt - Notepad

```
Sample source code: C:\Users\ASUS\Desktop\finalProject\DataSet\3\3 (13).cpp

Rank              Naive Bayes            K Means
  1                    4                   11
  2                    5                    6
  3                    3                    4
  4                    1                    3
  5                    2                    2
  6                    7                    1
  7                   16                   16
  8                   11                    5
  9                    6                   13
 10                   15                    7
```

- You will see a ranking of programmers (IDs) who are most likely to be the author of the anonymous source code. Results of Naïve-Bayes and K Means are presented together for better inference based on both of those results.

4. Conclusion

This project let me enter the realms of machine learning for the first time. Implementing Naïve-Bayes and K Means algorithm were fun and challenging at the same time. In a word, it has been an adventure to learn how statistics work with probability, inference and machine learning. I also learned how to parse through strings or lines in order to extract what I need from them. I learned a lot about how strings work and how I can do different manipulations on them. I handled such a large project for the first time and learned a lot about header files, connection between multiple source files etc. This project taught me how to represent data points using useful data structures and how to handle large number of them all together. It was a great experience and I can not be more thankful to my supervisor who constantly helped me wherever I struggled and led my path. I really look forward to using this experience in future projects and programming works.

5. Appendix

I got familiar with some other advanced machine learning algorithms (decision tree, random forest etc.) which I eagerly look forward to work on in the future.

References

1. https://en.wikipedia.org/wiki/K-means_clustering

2. https://en.wikipedia.org/wiki/Naive_Bayes_classifier

3.https://freedom-to-tinker.com/2015/01/21/anonymous-programmers-can-be-identified-by-analyzing-coding-style/

4.https://www.usenix.org/system/files/conference/usenixsecurity15/sec15-paper-caliskan-islam.pdf

5. https://en.wikipedia.org/wiki/Beta_distribution