Introduction

As part of our standard recruitment procedures, we ask data science candidates to undertake a small analysis exercise. The exercise consists of:

- Part 1: writing a program to solve a problem
- Part 2: analyzing and reporting on data
-

Please design and code this program alone. While you will have colleagues in your normal working environment, we need to see how good you are, not how good your current friends or colleagues are. If we hire you and your level of competence is not as indicated by this test, we will have to carefully consider whether your employment should continue past the probation period. If you do copy code from elsewhere, for any reason (e.g. a library routine), please clearly indicate its source.

All code, comments and documentation should be in English.

Tasks

1) A piece of malware is uniquely identified by its MD5. Multiple pieces of antivirus software can identify a piece of malware, and assign it a signature. Attached is a sample file, called *malware_hashes.csv*. It includes three columns:
    - **EN_ID** : The ID of the antivirus software
    - **SIGNATURE** : The signature assigned to a piece of malware by the antivirus software
    - **MD5** : The unique identifier of the malware

    A vulnerability is uniquely identified by its CVE ID, which follows the format : CVE-YYYY-NNNN, where YYYY is the calendar year the vulnerability was identified in, and NNNN is its ID number. If an antivirus software is able to identify the vulnerability a piece of malware exploits, it will include the CVE ID in the malware signature. For example, the signature “Java.Exploit.CVE-2009-3867.A” indicates that the malware exploits CVE-2009-3867. If a piece of malware exploits multiple vulnerabilities, antivirus software will provide multiple signatures, each with one CVE ID. For example, the malware identified by the MD5 “514d80e0ad5f9d793053414f7071638c” has multiple signatures, including “Java.Exploit.CVE-2010-0842.E”, “Java.Exploit.CVE-2012-0507.AI”, and “Java.Exploit.CVE-2013-0431.F”. This means that this piece of malware exploits (among others) CVE-2010-0842, CVE-2012-0507, and CVE-2013-0431.

    Given this information, write a python script *count_malware.*py that, given a CSV file in the format above, outputs a file with two columns, ‘cveid’ and ‘malware_count’ summarizing the number of unique pieces of malware for each CVE. The file should be sorted by the malware count, largest to smallest. It should run from the command line, with the input and output file names provided, for example :

    ```
    python count_malware.py malware_hashes.csv malware_count.csv
    ```

    For this section, you will be evaluated on correctness of output, efficiency of the algorithm, and adherence to common coding practices such as clear commenting and graceful error handling.

2) The other file in this folder, *vuln_data.csv*, provides Common Vulnerability Scoring System attributes for a variety of CVEs, some of which have malware associated with them. You should be able to identify which CVE-IDs have malware associated with them based on your answer to the previous problem. The Common Vulnerability Scoring System attributes provided are:
    - cvss_access_vector : 'N' if the vulnerability is network exploitable, 'A' if the vulnerability is adjacent network exploitable, and 'L' if the vulnerability is locally exploitable
    - cvss_access_complexity : 'L' if the vulnerability does not require any special circumstances to exploit, 'M' if the vulnerability requires somewhat special knowledge or circumstances to exploit, and 'H' if special knowledge or circumstances are required to exploit the vulnerability.
    - cvss_authentication : 'N' if no authentication is required to exploit the vulnerability, 'S' if an attacker must be logged in to exploit the vulnerability, and 'M' if an attacker must authenticate two or more times to exploit the vulnerability.
    - cvss_confidentiality_impact : 'N' if no confidential information will be exposed to an attacker who successfully exploits the vulnerability, 'P' if some confidential information will be exposed, and 'C' if the attacker can gain access to all confidential data.
    - cvss_integrity_impact : 'N' if a successful exploit of this vulnerability would not allow an attacker to modify any data, 'P' if an attacker could modify some data, and 'C' if an attacker could edit any files on the target system.
    - cvss_availability_impact : 'N' if a successful exploit of this vulnerability would not result in an interruption of service or decreased performance, 'P' if there could be a partial interruption or reduced performance, and 'C' if there would be a total shutdown of service

More information about these variables can be found here.

Using your answer to part (1) and the information in *vuln_data.csv,* write a paragraph detailing which factors appear to be related to malware correlation and which factors do not. Does anything surprise you? Conventional wisdom says that the access vector, authentication, and access complexity determine how exploitable a vulnerability is. Based on your analysis, does that appear to be true? Include your analysis in a file named *analysis.txt, and* any code used to analyze the data in a file named *analysis.py* (or *analysis.R* if you would prefer to use R).

For this section, you will be evaluated on depth of analysis and your ability to clearly communicate results of analysis to a non-technical audience.

Submission

You should submit your complete source code and the analysis document (please note the names required for the files) in a zip file with the name dataScience-PythonTest_<your name>.zip - for example, dataScience-PythonTest_ChiMo.zip. You should email it to [redacted] and [redacted] in the subject line "Data Science test results from <your name>".