BÁO CÁO LAB

Môn học: Phương pháp học máy trong an toàn thông tin

Tên chủ đề: Lab 2

GVHD: Nguyễn Hữu Quyền

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT522.O21.ATCL.1

STT	Họ và tên	MSSV	Email
1	Nguyễn Đại Nghĩa	21521182	21521182@gm.uit.edu.vn
2	Hoàng Gia Bảo	21521848	21521848@gm.uit.edu.vn
3	Trương Đặng Văn Linh	21520328	21520328@gm.uit.edu.vn
4	Mai Quốc Cường	21521901	21521901@gm.uit.edu.vn

Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

BÁO CÁO CHI TIẾT

Câu 1: Sinh viên so sánh kết quả băm với VirusTotal và website Python

```
[ ] import hashlib


    def hash_file(filename, hash_function):
        with open(filename, "rb") as f:
            data = f.read()
        hash_object = hash_function()
        hash_object.update(data)
        return hash_object.hexdigest()
```

```
[ ] filename = "./python-3.10.0-amd64.exe"
    md5_hash = hash_file(filename, hashlib.md5)
    sha256_hash = hash_file(filename, hashlib.sha256)

    print(f"Hash MD5: {md5_hash}")
    print(f"Hash SHA256: {sha256_hash}")
```

```
Hash MD5: c3917c08a7fe85db7203da6dcaa99a70
Hash SHA256: cb580eb7dc55f9198e650f016645023e8b2224cf7d033857d12880b46c5c94ef
```

KẾT QUẢ TỪ VIRUS TOTAL:

Basic properties

MD5	c3917c08a7fe85db7203da6dcaa99a70
SHA-1	3ee4e92a8ef94c70fb56859503fdc805d217d689
SHA-256	cb580eb7dc55f9198e650f016645023e8b2224cf7d033857d12880b46c5c94ef
Vhash	027066655d15156562d5z600967z8041z11z32z17fz
Authentihash	d2ac499a07a14271a405a98da7ce8c9e51b344a5073c7821dd2b11cc3eeb7edb
Imphash	d7e2fd259780271687ffca462b9e69b7
Rich PE header hash	8f80f40ebeaa258c069d4bca6f75473c
SSDEEP	786432:oRGpDzOulaFdKz/NHtWO6PvbKAhjHC03fGeRj:JpDaulaFdKz/ltZbAhe0PGe
TLSH	T1E05733332065D472E5E00273F9187534BE78AE2851504CAEE7D8FD6A1BB84637BF7A81
File type	Win32 EXE executable windows win32 pe peexe
Magic	PE32 executable (GUI) Intel 80386, for MS Windows
TrID	Windows Control Panel Item (generic) (68.8%) | Win64 Executable (generic) (12.5%) | Win16 NE executable (generic) (6%) | Win32 Executable (generic) (5.3%) | OS/…
DetectItEasy	PE32 | Installer: WiX Toolset installer (1.0) | Archive: Microsoft Cabinet File (1.03) [MSZip,35.6%,6 files] | Compiler: Microsoft Visual C/C++ (19.11.25508) [C++] | Linke…
File size	27.00 MB (28315928 bytes)

Có thể thấy kết quả từ Python script và Virus Total là giống nhau, từ đó cho thấy script đã hoạt động tốt việc hash file với MD5 và SHA256.

Câu 2: Sinh viên cho biết quả của đoạn code

```python
import pefile
import os

directories = ["Benign PE Samples", "Malicious PE Samples"]


def get_section_names(pe):
    list_of_section_names = []
    for sec in pe.sections:
        normalized_name = sec.Name.decode().replace("\x00", "").lower()
        list_of_section_names.append(normalized_name)

    return list_of_section_names


def preprocess_imports(list_of_DLLs):
    return [x.decode().split(".")[0].lower() for x in list_of_DLLs]


def get_imports(pe):
    list_of_imports = []
    for entry in pe.DIRECTORY_ENTRY_IMPORT:
        list_of_imports.append(entry.dll)
    return preprocess_imports(list_of_imports)
```

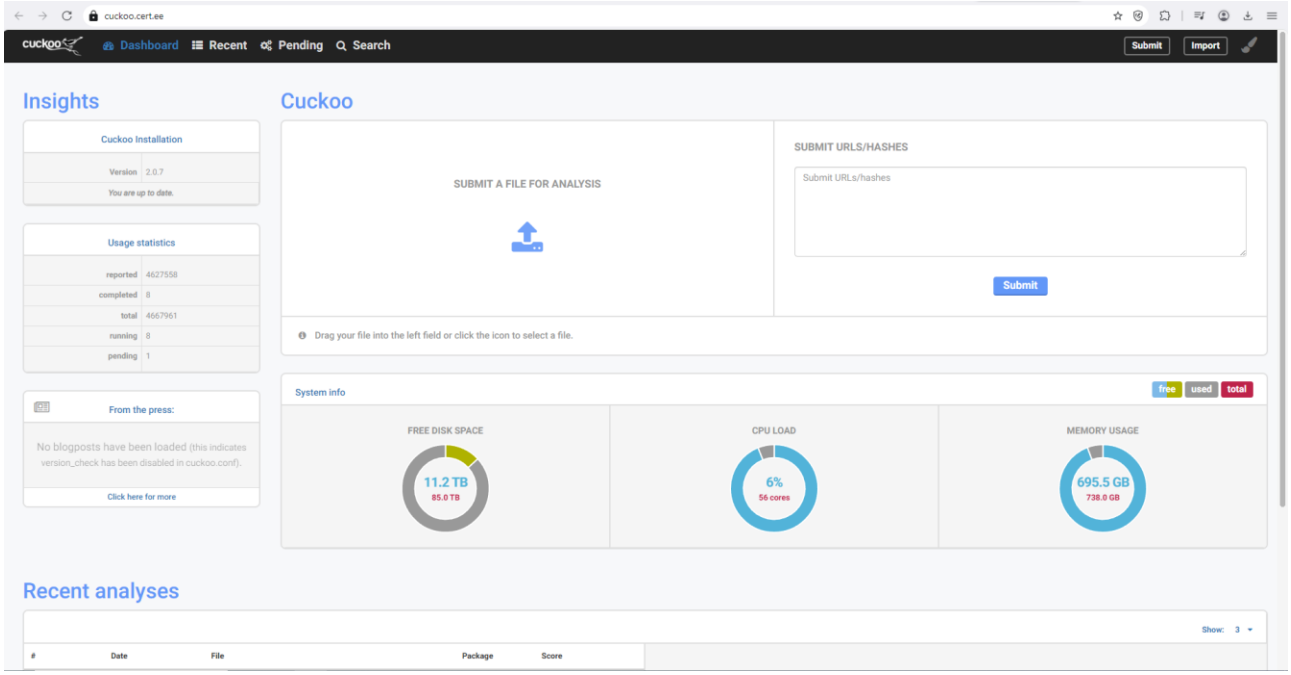
```
[ ] for dataset in directories:
        samples = []
        importCorpus = []
        numSection = []
        sectionName = []
        for file in os.listdir(dataset):
            fileDir = os.path.join(dataset, file)
            if os.path.isfile(fileDir):
                samples.append(fileDir)

        for file in samples:
            try:
                pe = pefile.PE(file)

                imports = get_imports(pe)
                n_sections = len(pe.sections)
                sec_names = get_section_names(pe)

                importCorpus.append(imports)
                numSection.append(n_sections)
                sectionName.append(sec_names)

            except Exception as e:
                print(f"{e} for file {file}")

        print(f"------{dataset}------")
        print(f"Import corpus: {importCorpus}")
        print(f"Number of sections: {numSection}")
        print(f"Section name: {sectionName}")
        print()
```

Kết quả như sau:

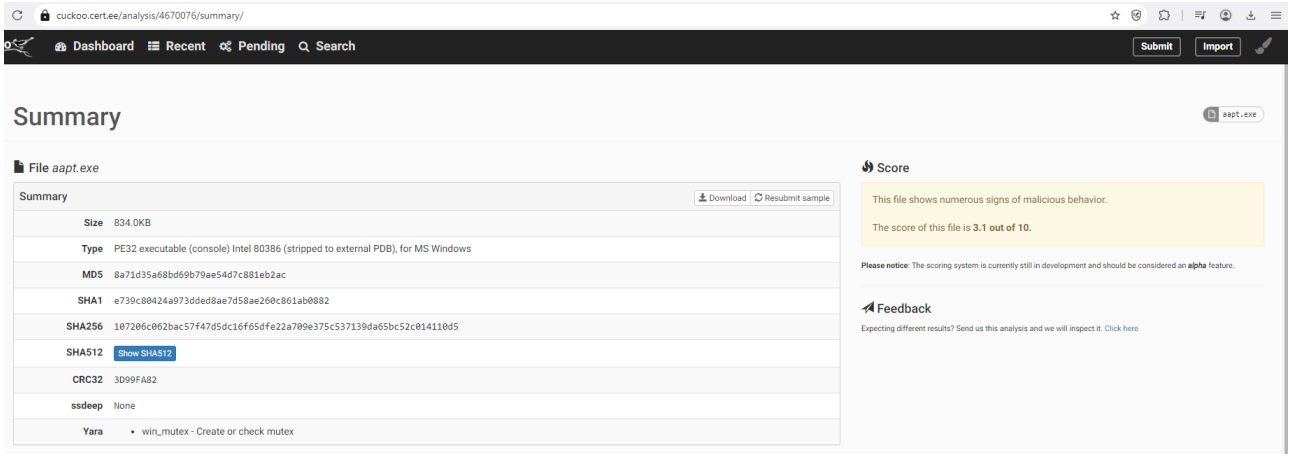
```
'DOS Header magic not found.' for file Benign PE Samples/hvsirdpclient.exe
'DOS Header magic not found.' for file Benign PE Samples/hvsirpcd.exe
'DOS Header magic not found.' for file Benign PE Samples/iisreset.exe
'DOS Header magic not found.' for file Benign PE Samples/iisrstas.exe
'DOS Header magic not found.' for file Benign PE Samples/iissetup.exe
'DOS Header magic not found.' for file Benign PE Samples/inetinfo.exe
'DOS Header magic not found.' for file Benign PE Samples/InetMgr.exe
'DOS Header magic not found.' for file Benign PE Samples/InetMgr6.exe
'DOS Header magic not found.' for file Benign PE Samples/InspectVhdDialog.exe
'DOS Header magic not found.' for file Benign PE Samples/InspectVhdDialog6.2.exe
'DOS Header magic not found.' for file Benign PE Samples/InspectVhdDialog6.3.exe
'DOS Header magic not found.' for file Benign PE Samples/ldifde.exe
'DOS Header magic not found.' for file Benign PE Samples/ldp.exe
'DOS Header magic not found.' for file Benign PE Samples/LogCollector.exe
'DOS Header magic not found.' for file Benign PE Samples/lpq.exe
'DOS Header magic not found.' for file Benign PE Samples/lpr.exe
'DOS Header magic not found.' for file Benign PE Samples/lxRun.exe
------Benign PE Samples------
Import corpus: [['api-ms-win-crt-runtime-l1-1-0', 'api-ms-win-crt-string-l1-1-0', 'api-ms-win-crt-private-l1-1-0', 'api-ms-win-core-libraryloader-l1-2-0', 'api-ms-win-core-synch-l1-1-0', 'api-ms-win-core-heap-l1-1-0', 'api-ms-win-cor
Number of sections: [6, 7, 7, 7, 7, 5, 6, 5, 7, 5, 5, 5, 5, 5, 5, 5, 5, 5, 7, 6, 5, 6, 6, 6, 6, 5, 6, 7, 5, 6, 5, 7, 3, 5, 5, 6, 6, 5, 5, 3, 4, 6, 6, 5, 5, 6, 8, 5, 5, 7, 7, 7, 7, 6, 6, 7, 6, 5, 5, 5, 6, 6, 6, 6, 6, 10, 7, 7, 5, 5]
Section name: [['.text', '.rdata', '.data', '.pdata', '.didat', '.reloc'], ['.text', '.rdata', '.data', '.pdata', '.didat', '.rsrc', '.reloc'], ['.text', '.rdata', '.data', '.pdata', '.didat', '.rsrc', '.reloc'], ['.text', '.rdata',

------Malicious PE Samples------
Import corpus: [['kernel32', 'msvcrt', 'msvcrt']]
Number of sections: [5]
Section name: [['.text', '.data', '.rdata', '.bss', '.idata']]
```

Câu 3: Sinh viên tự tìm hiểu, cài đặt cuckoo, thực hiện và trình bày phân tích động một tập tin PE

Bởi vì gặp một số trục trặc khi cài đặt cuckoo bằng máy ảo, nên nhóm đã sử dụng cuckoo online để thực hiện phân tích file aapt.exe
Màn hình khi truy cập vào trang web:



Sau khi file được phân tích thành công thì kết quả có được như sau:



Information on Execution

Analysis					
Category	Started	Completed	Duration	Routing	Logs
FILE	April 15, 2024, 6:38 p.m.	April 15, 2024, 6:39 p.m.	47 seconds	internet	Show Analyzer Log Show Cuckoo Log

Signatures

Yara rule detected for file (1 event)

description	Create or check mutex	rule	win_mutex

Command line console output was observed (18 events)

Time & API	Arguments	Status	Return	Repeated
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: A console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: ndroid Asset Packaging Tool console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: U console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: sage: console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: aapt l[ist] [-v] [-a] file.{zip,jar,apk} List contents of Zip-compatible archive. console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: aapt d[ump] [--values] WHAT file.{apk} [asset [asset ...]] badging Print the label and icon for the app declared in APK. permissions Print the permissions from the APK. resources Print the resource table from the APK. configurations Print the configurations in the APK. xmltree Print the compiled xmls in the given assets. xmlstrings Print the strings of the given compiled xml assets. console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: aapt p[ackage] [-d][-f][-m][-u][-v][-x][-z][-M AndroidManifest.xml] \ [-0 extension [-0 extension ...]] [-g tolerance] [-j jarfile] \ [--debug-mode] [--min-sdk-version VAL] [--target-sdk-version VAL] \ [--app-version VAL] [--app-version-name TEXT] [--custom-package VAL] \ [--rename-manifest-package PACKAGE] \ [--rename-instrumentation-target-package PACKAGE] \ [--utf16] [--auto-add-overlay] \ [--max-res-version VAL] \ [-I base-package [-I base-package ...]] \ [-A asset-source-dir] [-G class-list-file] [-P public-definitions-file] \ [-S resource-sources [-S resource-sources ...]] \ [-F apk-file] [-J R-file-dir] \ [--product product1,product2,...] \ [-c CONFIGS] [--preferred-configurations CONFIGS] \ [raw-files-dir [raw-files-dir] ...] \ [--output-text-symbols DIR] Package the android resources. It will read assets and resources that are supplied with the -M -A -S o console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: r raw-files-dir arguments. The -J -P -F and -R options control which files are output. console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: aapt r[emove] [-v] file.{zip,jar,apk} file1 [file2 ...] Delete specified files from Zip-compatible archive. console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: aapt a[dd] [-v] file.{zip,jar,apk} file1 [file2 ...] Add specified files to Zip-compatible archive. console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: aapt c[runch] [-v] -S resource-sources ... -C output-folder ... Do PNG preprocessing and store the results in output folder. console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: aapt v[ersion] Print program version. console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: Modifiers: -a print Android-specific data (resources, manifest) when listing -c specify which configurations to include. The default is all configurations. The value of the parameter should be a comma separated list of configuration values. Locales should be specified as either a language or language-region pair. Some examples: en port,en port,land,en_US If you put the special locale, zz_ZZ on the list, it will perform pseudolocalization on the default locale, modifying all of the strings so you can look for strings that missed the internationalization process. For example: port,land,zz_ZZ -d one or more device assets to include, separated by commas -f force overwrite of existing files -g specify a pixel tolerance to force images to grayscale, default 0 -j specify a jar or zip file containing classes to include -k junk path of file(s) added -m make package d console_handle: 0x0000000b	1	1	0

WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: irectories under location specified by -J -u update existing packages (add new, replace older, remove deleted files) -v verbose output -x create extending (non-application) resource IDs -z require localization of resource attributes marked with localization="suggested" -A additional directory in which to find raw asset files -G A file to output proguard options into. -F specify the apk file to output -I add an existing package to base include set -J specify where to output R.java resource constant definitions -M specify full path to AndroidManifest.xml to include in zip -P specify where to output public resource definitions -S directory in which to find resources. Multiple directories will be scanned and the first match found (left to right) will take precedence. -0 specifies an additional extension for which such files will not be stored compressed in the .apk. An empty string means to not compress any files at console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: all. --debug-mode inserts android:debuggable="true" in to the application node of the manifest, making the application debuggable even on production devices. --min-sdk-version inserts android:minSdkVersion in to manifest. If the version is 7 or higher, the default encoding for resources will be in UTF-8. --target-sdk-version inserts android:targetSdkVersion in to manifest. --max-res-version ignores versioned resource directories above the given value. --values when used with "dump resources" also includes resource values. --version-code inserts android:versionCode in to manifest. --version-name inserts android:versionName in to manifest. --custom-package generates R.java into a different package. --extra-packages generate R.java for libraries. Separate libraries with ':'. --generate-dependencies generate dependency files in the same directories for R.java and resource packa console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: ge --auto-add-overlay Automatically add resources that are only in overlays. --preferred-configurations Like the -c option for filtering out unneeded configurations, but only expresses a preference. If there is no resource available with the preferred configuration then it will not be stripped. --rename-manifest-package Rewrite the manifest so that its package name is the package name given here. Relative class names (for example .Foo) will be changed to absolute names with the old package so that the code does not need to change. --rename-instrumentation-target-package Rewrite the manifest so that all of its instrumentation components target the given package. Useful when used in conjunction with --rename-manifest-package to fix tests against a package that has been renamed. --product Specifies which variant to choose for strings that have product variants --utf16 console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: changes default encoding for resources to UTF-16. Only useful when API console_handle: 0x0000000b	1	1	0
WriteConsoleA April 15, 2024, 7:38 p.m.	buffer: level is set to 7 or higher where the default encoding is UTF-8. --non-constant-id Make the resources ID non constant. This is required to make an R java class that does not contain the final value but is used to make reusable compiled libraries that need to access resources. --error-on-failed-insert Forces aapt to return an error if it fails to insert values into the manifest with --debug-mode, --min-sdk-version, --target-sdk-version --version-code and --version-name. Insertion typically fails if the manifest already defines the attribute. --output-text-symbols Generates a text file containing the resource symbols of the R class in the specified folder. --ignore-assets Assets to be ignored. Default pattern is: !.svn:!.git:!.ds_store:!*.scc:.*: <dir>_*:!CVS:!thumbs.db:!picasa.ini:!*~ console_handle: 0x0000000b	1	1	0

Câu 4: Tương tự sinh viên hãy làm các câu truy vấn về Python và Powershell

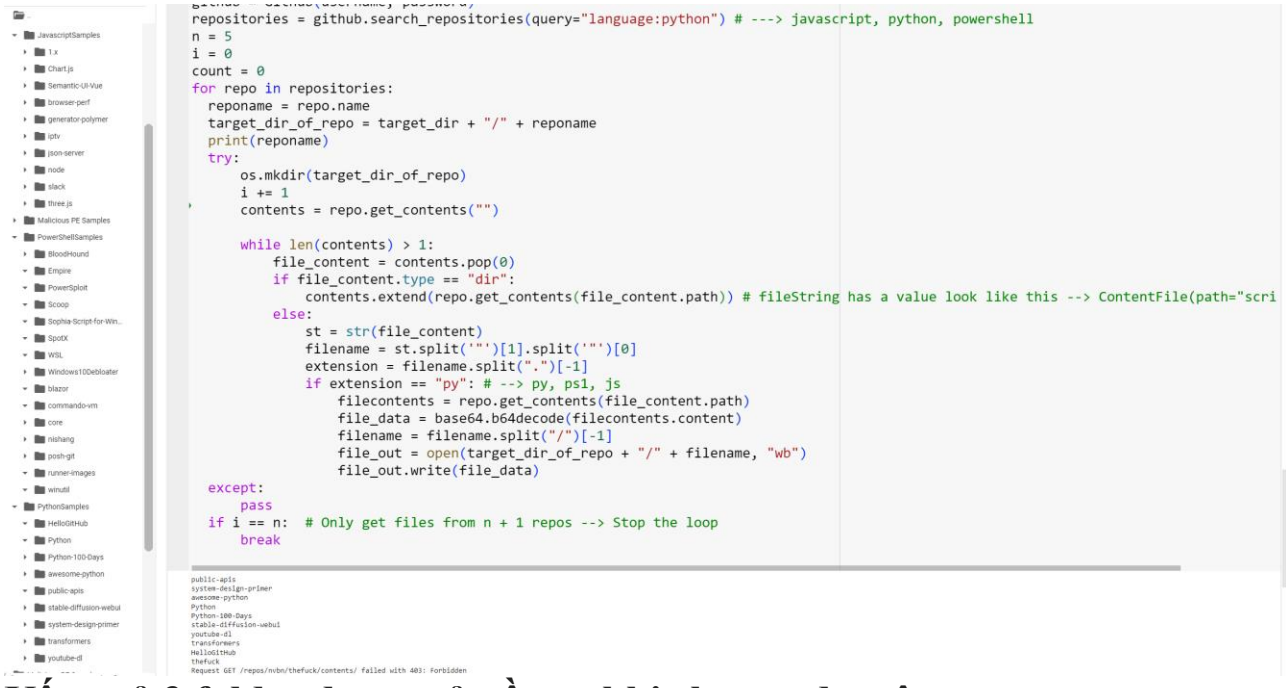
SCRIPT CODE

```
import os
from github import Github
import base64
```

POWERSHELL

```
[5] username = "GBao294"
    password = "UnHbNod4im5nPjurBs82gShjSjgfqg5Fk9ofLKcvn0w" # Account token, only repo's read permission required.
    target_dir = "/content/LAB2/PowerShellSamples" # ---> Output dir to save resulting files ( JavascriptSamples, PythonSamples, PowerShellSamples)
    github = Github(username, password)
    repositories = github.search_repositories(query="language:powershell") # ---> javascript, python, powershell
    n = 5
    i = 0
    count = 0
```

```
[7] for repo in repositories:
      reponame = repo.name
      target_dir_of_repo = target_dir + "/" + reponame
      print(reponame)
      try:
          os.mkdir(target_dir_of_repo)
          i += 1
          contents = repo.get_contents("")

          while len(contents) > 1:
              file_content = contents.pop(0)
              if file_content.type == "dir":
                  contents.extend(repo.get_contents(file_content.path)) # fileString has a value look like this --> ContentFile(path="script.js")
              else:
                  st = str(file_content)
                  filename = st.split('"')[1].split('"')[0]
                  extension = filename.split(".")[-1]
                  if extension == "ps1": # --> py, ps1, js
                      filecontents = repo.get_contents(file_content.path)
                      file_data = base64.b64decode(filecontents.content)
                      filename = filename.split("/")[-1]
                      file_out = open(target_dir_of_repo + "/" + filename, "wb")
                      file_out.write(file_data)
      except:
          pass
      if i == n:  # Only get files from n + 1 repos --> Stop the loop
          break
```



```
repositories = github.search_repositories(query="language:python") # ---> javascript, python, powershell
n = 5
i = 0
count = 0
for repo in repositories:
  reponame = repo.name
  target_dir_of_repo = target_dir + "/" + reponame
  print(reponame)
  try:
      os.mkdir(target_dir_of_repo)
      i += 1
      contents = repo.get_contents("")

      while len(contents) > 1:
          file_content = contents.pop(0)
          if file_content.type == "dir":
              contents.extend(repo.get_contents(file_content.path)) # fileString has a value look like this --> ContentFile(path="scri
          else:
              st = str(file_content)
              filename = st.split('"')[1].split('"')[0]
              extension = filename.split(".")[-1]
              if extension == "py": # --> py, ps1, js
                  filecontents = repo.get_contents(file_content.path)
                  file_data = base64.b64decode(filecontents.content)
                  filename = filename.split("/")[-1]
                  file_out = open(target_dir_of_repo + "/" + filename, "wb")
                  file_out.write(file_data)
  except:
      pass
  if i == n:  # Only get files from n + 1 repos --> Stop the loop
      break
```

```
public-apis
system-design-primer
awesome-python
Python
Python-100-Days
stable-diffusion-webui
youtube-dl
transformers
HelloGitHub
thefuck
Request GET /repos/nvbn/thefuck/contents/ failed with 403: Forbidden
```

Kết quả 3 folder được trả về sau khi chạy code trên

Câu 5: Sinh viên cho biết quả của đoạn code

Bt5 Ex5: Classification by Extension

```
import os
from sklearn.feature_extraction.text import HashingVectorizer, TfidfTransformer
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.pipeline import Pipeline
```

```
[17] javascript_path = "/content/LAB2/JavascriptSamples/"
     python_path = "/content/LAB2/PythonSamples/"
     powershell_path = "/content/LAB2/PowerShellSamples/"
```

```
[18] corpus = []
     labels = []
     file_types_and_labels = [(javascript_path, -1), (python_path, 0), (powershell_path, 1)]
```

```
for files_path, label in file_types_and_labels:
  folder = os.listdir(files_path)
  for subfolder in folder:
    print("===== USING FOLDER: %s =====" % subfolder)
    path = files_path + subfolder
    for file in os.listdir(path):
      print("- Running at file: %s" % (path + '/' + file))
      data = ''

      try:
        with open(f'{path}/{file}', 'r') as myfile:
          data = myfile.read().replace('\n', '')


      except:
        pass
      data = str(data)
      corpus.append(data)
      labels.append(label)
```

```python
for files_path, label in file_types_and_labels:
  folder = os.listdir(files_path)
  for subfolder in folder:
    print("===== USING FOLDER: %s =====" % subfolder)
    path = files_path + subfolder
    for file in os.listdir(path):
      print("- Running at file: %s" % (path + '/' + file))
      data = ''

      try:
        with open(f'{path}/{file}', 'r') as myfile:
          data = myfile.read().replace('\n', '')


      except:
        pass
      data = str(data)
      corpus.append(data)
      labels.append(label)
```

```
===== USING FOLDER: Chart.js =====
===== USING FOLDER: json-server =====
===== USING FOLDER: three.js =====
===== USING FOLDER: node =====
===== USING FOLDER: slack =====
===== USING FOLDER: browser-perf =====
===== USING FOLDER: Semantic-UI-Vue =====
===== USING FOLDER: 1.x =====
===== USING FOLDER: generator-polymer =====
===== USING FOLDER: iptv =====
===== USING FOLDER: HelloGitHub =====
===== USING FOLDER: public-apis =====
===== USING FOLDER: thefuck =====
===== USING FOLDER: system-design-primer =====
===== USING FOLDER: stable-diffusion-webui =====
===== USING FOLDER: transformers =====
===== USING FOLDER: django =====
===== USING FOLDER: langchain =====
===== USING FOLDER: youtube-dl =====
===== USING FOLDER: Python-100-Days =====
===== USING FOLDER: Python =====
===== USING FOLDER: models =====
===== USING FOLDER: awesome-python =====
===== USING FOLDER: pytorch =====
===== USING FOLDER: blazor =====
===== USING FOLDER: runner-images =====
===== USING FOLDER: PowerSploit =====
===== USING FOLDER: BloodHound =====
===== USING FOLDER: winutil =====
===== USING FOLDER: Sophia-Script-for-Windows =====
===== USING FOLDER: Windows10Debloater =====
===== USING FOLDER: nishang =====
===== USING FOLDER: Empire =====
===== USING FOLDER: WSL =====
===== USING FOLDER: posh-git =====
===== USING FOLDER: commando-vm =====
```

```python
      print("- Running at file: %s" % (path + '/' + file))
      data = ''

      try:
        with open(f'{path}/{file}', 'r') as myfile:
          data = myfile.read().replace('\n', '')


      except:
        pass
      data = str(data)
      corpus.append(data)
      labels.append(label)
```

```
- Running at file: /content/PythonSamples/PythonSamples\Python/gray_code_sequence.py
- Running at file: /content/PythonSamples/PythonSamples\Python/power_sum.py
- Running at file: /content/PythonSamples/PythonSamples\Python/binary_to_octal.py
- Running at file: /content/PythonSamples/PythonSamples\Python/swap_all_odd_and_even_bits.py
- Running at file: /content/PythonSamples/PythonSamples\Python/energy_conversions.py
- Running at file: /content/PythonSamples/PythonSamples\Python/a1z26.py
- Running at file: /content/PythonSamples/PythonSamples\Python/nor_gate.py
- Running at file: /content/PythonSamples/PythonSamples\Python/rot13.py
- Running at file: /content/PythonSamples/PythonSamples\Python/excel_title_to_column.py
- Running at file: /content/PythonSamples/PythonSamples\Python/bitwise_addition_recursive.py
- Running at file: /content/PythonSamples/PythonSamples\Python/is_even.py
- Running at file: /content/PythonSamples/PythonSamples\Python/mixed_keyword_cypher.py
- Running at file: /content/PythonSamples/PythonSamples\Python/decimal_to_any.py
- Running at file: /content/PythonSamples/PythonSamples\Python/count_number_of_one_bits.py
- Running at file: /content/PythonSamples/PythonSamples\Python/base16.py
- Running at file: /content/PythonSamples/PythonSamples\Python/affine_cipher.py
- Running at file: /content/PythonSamples/PythonSamples\Python/rsa_factorization.py
- Running at file: /content/PythonSamples/PythonSamples\Python/deterministic_miller_rabin.py
- Running at file: /content/PythonSamples/PythonSamples\Python/generate_parentheses.py
- Running at file: /content/PythonSamples/PythonSamples\Python/__init__.py
- Running at file: /content/PythonSamples/PythonSamples\Python/rabin_miller.py
- Running at file: /content/PythonSamples/PythonSamples\Python/nimply_gate.py
- Running at file: /content/PythonSamples/PythonSamples\Python/binary_coded_decimal.py
- Running at file: /content/PythonSamples/PythonSamples\Python/fractionated_morse_cipher.py
- Running at file: /content/PythonSamples/PythonSamples\Python/langtons_ant.py
- Running at file: /content/PythonSamples/PythonSamples\Python/knight_tour.py
- Running at file: /content/PythonSamples/PythonSamples\Python/decimal_to_hexadecimal.py
- Running at file: /content/PythonSamples/PythonSamples\Python/atbash.py
- Running at file: /content/PythonSamples/PythonSamples\Python/mosaic_augmentation.py
- Running at file: /content/PythonSamples/PythonSamples\Python/length_conversion.py
- Running at file: /content/PythonSamples/PythonSamples\Python/rsa_key_generator.py
- Running at file: /content/PythonSamples/PythonSamples\Python/excess_3_code.py
- Running at file: /content/PythonSamples/PythonSamples\Python/conways_game_of_life.py
- Running at file: /content/PythonSamples/PythonSamples\Python/wa_tor.py
- Running at file: /content/PythonSamples/PythonSamples\Python/transposition_cipher_encrypt_decrypt_file.py
- Running at file: /content/PythonSamples/PythonSamples\Python/numbers_different_signs.py
- Running at file: /content/PythonSamples/PythonSamples\Python/autokey.py
- Running at file: /content/PythonSamples/PythonSamples\Python/binary_to_decimal.py
- Running at file: /content/PythonSamples/PythonSamples\Python/hex_to_bin.py
- Running at file: /content/PythonSamples/PythonSamples\Python/elgamal_key_generator.py
- Running at file: /content/PythonSamples/PythonSamples\Python/butterworth_filter.py
- Running at file: /content/PythonSamples/PythonSamples\Python/brute_force_caesar_cipher.py
- Running at file: /content/PythonSamples/PythonSamples\Python/diffie_hellman.py
- Running at file: /content/PythonSamples/PythonSamples\Python/rgb_hsv_conversion.py
```

Kết quả

```
y_test_pred = text_clf.predict(X_test)
print(accuracy_score(y_test, y_test_pred))
print(confusion_matrix(y_test, y_test_pred)) # Print Confusion Matrix
```

```
0.958041958041958
[[ 48   2   2]
 [  0  63   1]
 [  0   7 163]]
```

Kết quả

Câu 6: Sinh viên cho biết quả của đoạn code

Đầu tiên là em sẽ thực hiện đo lường sự tương đồng giữa 4 chuỗi theo như hướng dẫn:

```
import ssdeep
str1 = "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua."
str2 = "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore Magna aliqua."
str3 = "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore aliqua."
str4 = "Something completely different from the other strings."

hash1 = ssdeep.hash(str1)
hash2 = ssdeep.hash(str2)
hash3 = ssdeep.hash(str3)
hash4 = ssdeep.hash(str4)
print(hash1)
print(hash2)
print(hash3)
print(hash4)

print(ssdeep.compare(hash1, hash1))
print(ssdeep.compare(hash1, hash2))
print(ssdeep.compare(hash1, hash3))
print(ssdeep.compare(hash1, hash4))
```

Nhưng mà để chạy được đoạn code trên thì em cũng phải tiến hành cài đặt ssdeep cùng với dependency của nó:

```
[6] !apt install libfuzzy-dev
```

```
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libfuzzy2
The following NEW packages will be installed:
  libfuzzy-dev libfuzzy2
0 upgraded, 2 newly installed, 0 to remove and 45 not upgraded.
Need to get 27.8 kB of archives.
After this operation, 104 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libfuzzy2 amd64 2.14.1+git20180629.57fcfff-2 [14.8 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libfuzzy-dev amd64 2.14.1+git20180629.57fcfff-2 [13.0 kB]
Fetched 27.8 kB in 0s (134 kB/s)
Selecting previously unselected package libfuzzy2:amd64.
(Reading database ... 121752 files and directories currently installed.)
Preparing to unpack .../libfuzzy2_2.14.1+git20180629.57fcfff-2_amd64.deb ...
Unpacking libfuzzy2:amd64 (2.14.1+git20180629.57fcfff-2) ...
Selecting previously unselected package libfuzzy-dev:amd64.
Preparing to unpack .../libfuzzy-dev_2.14.1+git20180629.57fcfff-2_amd64.deb ...
Unpacking libfuzzy-dev:amd64 (2.14.1+git20180629.57fcfff-2) ...
Setting up libfuzzy2:amd64 (2.14.1+git20180629.57fcfff-2) ...
Setting up libfuzzy-dev:amd64 (2.14.1+git20180629.57fcfff-2) ...
Processing triggers for libc-bin (2.35-0ubuntu3.4) ...
/sbin/ldconfig.real: /usr/local/lib/libtbb.so.12 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_5.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc_proxy.so.2 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc.so.2 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_0.so.3 is not a symbolic link
```

```
!pip3 install ssdeep
```

```
Collecting ssdeep
  Using cached ssdeep-3.4.tar.gz (110 kB)
  Preparing metadata (setup.py) ... done
Requirement already satisfied: cffi>=0.8.6 in /usr/local/lib/python3.10/dist-packages (from ssdeep) (1.16.0)
Requirement already satisfied: six>=1.4.1 in /usr/local/lib/python3.10/dist-packages (from ssdeep) (1.16.0)
Requirement already satisfied: pycparser in /usr/local/lib/python3.10/dist-packages (from cffi>=0.8.6->ssdeep) (2.22)
Building wheels for collected packages: ssdeep
  Building wheel for ssdeep (setup.py) ... done
  Created wheel for ssdeep: filename=ssdeep-3.4-cp310-cp310-linux_x86_64.whl size=33290 sha256=f9ee5f93d51be7b5dd1cb2b23d14681966f9aa8beb6c84e01a771fac168707c1
  Stored in directory: /root/.cache/pip/wheels/29/9c/cb/04794e9a89fdec3acfb67930f12e99a727d1159f042b713c03
Successfully built ssdeep
Installing collected packages: ssdeep
Successfully installed ssdeep-3.4
```

Kết quả chạy được là:

```
3:f4oo8MRwRJFGW1gC6uWv6MQ2MFSl+JuBF8BSnJi:f4kPvtHMCMubyFtQ
3:f4oo8MRwRJFGW1gC6uWv6MQ2MFSl+JuBF8BS+EFECJi:f4kPvtHMCMubyFIsJQ
3:f4oo8MRwRJFGW1gC6uWv6MQ2MFSl+JuBF8BS6:f4kPvtHMCMubyF0
3:60QKZ+4CDTfDaRFKYLVL:ywKDC2mVL
100
39
37
0
```

Tiếp đến là em sẽ thực hiện việc đo lường mức độ giống nhau giữa hai tập tin như trong hướng dẫn.
Bước đầu là sẽ thêm vài byte null vào file python-3.10.0-amd64-fake.exe:

```
File  Actions  Edit  View  Help
┌──(nghianguyen㉿kali)-[~/Machine Learning attt]
└─$ truncate -s +1 python-3.10.0-amd64-fake.exe

┌──(nghianguyen㉿kali)-[~/Machine Learning attt]
└─$
```

Sau đó chạy hexdump để xem sự khác nhau giữa hai tập tin trước và sau. Đây là trước khi thêm vài null bytes:

```
nghianguyen@kali: ~/Machine Learning
File  Actions  Edit  View  Help

┌──(nghianguyen㉿kali)-[~/Machine Learning attt]
└─$ hexdump -C python-3.10.0-amd64.exe | tail -5
01b010e0  10 9c 34 66 02 d3 51 8c  b1 64 19 f3 55 12 0e 74  |..4f..Q..d..U..t|
01b010f0  38 71 4c 2e 1c db 44 d4  f3 81 31 a5 9c 2e c6 06  |8qL...D...1.....|
01b01100  4f 33 c6 8a 9a 5e 16 52  8c 4b 55 10 2b cd 45 61  |O3...^.R.KU.+.Ea|
01b01110  a5 00 00 00 00 00 00 00                           |........|
01b01118

┌──(nghianguyen㉿kali)-[~/Machine Learning attt]
└─$ hexdump -C python-3.10.0-amd64-fake.exe | tail -5
01b010e0  10 9c 34 66 02 d3 51 8c  b1 64 19 f3 55 12 0e 74  |..4f..Q..d..U..t|
01b010f0  38 71 4c 2e 1c db 44 d4  f3 81 31 a5 9c 2e c6 06  |8qL...D...1.....|
01b01100  4f 33 c6 8a 9a 5e 16 52  8c 4b 55 10 2b cd 45 61  |O3...^.R.KU.+.Ea|
01b01110  a5 00 00 00 00 00 00 00                           |........|
01b01118

┌──(nghianguyen㉿kali)-[~/Machine Learning attt]
└─$
```

Sau khi thêm vài null bytes:

```
┌──(nghianguyen㉿kali)-[~/Machine Learning attt]
└─$ hexdump -C python-3.10.0-amd64.exe | tail -5
01b010e0  10 9c 34 66 02 d3 51 8c  b1 64 19 f3 55 12 0e 74  |..4f..Q..d..U..t|
01b010f0  38 71 4c 2e 1c db 44 d4  f3 81 31 a5 9c 2e c6 06  |8qL...D...1.....|
01b01100  4f 33 c6 8a 9a 5e 16 52  8c 4b 55 10 2b cd 45 61  |O3...^.R.KU.+.Ea|
01b01110  a5 00 00 00 00 00 00 00                           |........|
01b01118

┌──(nghianguyen㉿kali)-[~/Machine Learning attt]
└─$ hexdump -C python-3.10.0-amd64-fake.exe | tail -5
01b010e0  10 9c 34 66 02 d3 51 8c  b1 64 19 f3 55 12 0e 74  |..4f..Q..d..U..t|
01b010f0  38 71 4c 2e 1c db 44 d4  f3 81 31 a5 9c 2e c6 06  |8qL...D...1.....|
01b01100  4f 33 c6 8a 9a 5e 16 52  8c 4b 55 10 2b cd 45 61  |O3...^.R.KU.+.Ea|
01b01110  a5 00 00 00 00 00 00 00  00                      |.........|
01b01119
```

Thông qua kết quả trên có thể thấy được rằng là đã có sự khác nhau.

Cuối cùng là em sử dụng ssdeep để so sánh 2 tập tin:

```
import ssdeep
hash1 = ssdeep.hash_from_file("/content/drive/MyDrive/python-3.10.0-amd64.exe")
hash2 = ssdeep.hash_from_file("/content/drive/MyDrive/python-3.10.0-amd64-fake.exe")
ssdeep.compare(hash1, hash2)
```

Kết quả nhận được là:

```
100
```

Vậy có thể thấy được là sử dụng ssdeep thì nó không thể phát hiện ra sự khác nhau khi thêm 1 vài bytes null.

Câu 7: Sinh viên cho biết quả của đoạn code

Bước đầu tiên em sẽ thực hiện việc trích xuất N-grams thông qua việc sử dụng code như hướng dẫn trong phần “i) Trích xuất N-grams” như sau:

```python
import collections
from nltk import ngrams

file_to_analyze = "/content/drive/MyDrive/python-3.10.0-amd64.exe"

def read_file(file_path):
  """Reads in the binary sequence of a binary file."""
  with open(file_path, "rb") as binary_file:
    data = binary_file.read()
  return data

def byte_sequence_to_Ngrams(byte_sequence, N):
 """Creates a list of N-grams from a byte sequence."""
 Ngrams = ngrams(byte_sequence, N)
 return list(Ngrams)

def binary_file_to_Ngram_counts(file, N):
 """Takes a binary file and outputs the N-grams counts of its binary sequence."""
 filebyte_sequence = read_file(file)
 file_Ngrams = byte_sequence_to_Ngrams(filebyte_sequence, N)
 return collections.Counter(file_Ngrams)

extracted_Ngrams = binary_file_to_Ngram_counts(file_to_analyze, 4)
print(extracted_Ngrams.most_common(10))
```

Giải thích sơ qua đoạn code trên thì việc đầu tiên là nó nhận đường dẫn đến một tệp nhị phân và đọc nội dung của tệp đó. Nó mở tệp ở chế độ đọc nhị phân, dữ liệu trả về là một chuỗi byte của tệp.
Tiếp đến là chuyển đổi một chuỗi byte thành một danh sách các n-gram. Sử dụng hàm ngrams từ thư viện nltk, hàm này tạo các n-gram có độ dài N từ chuỗi byte đầu vào. Mỗi n-gram là một tuple gồm N byte liên tiếp. Sau đó trả về danh sách các n-gram này.
Cuối cùng, nó sử dụng lớp Counter từ module collections để đếm số lần xuất hiện của từng n-gram. Kết quả trả về là một đối tượng Counter mô tả số lượng mỗi n-gram.
Mục đích chính của toàn bộ code là tạo các n-gram gồm 4 byte. Sau đó, nó in ra 10 n-gram phổ biến nhất và số lần xuất hiện của chúng.

Sau khi chạy đoạn code trên thì kết quả nhận lại được như sau:

```
[((0, 0, 0, 0), 24290), ((139, 240, 133, 246), 1920), ((32, 116, 111, 32), 1791), ((255, 255, 255, 255), 1671),
 ((108, 101, 100, 32), 1522), ((100, 32, 116, 111), 1519), ((97, 105, 108, 101), 1513), ((105, 108, 101, 100), 1513),
 ((70, 97, 105, 108), 1505), ((101, 100, 32, 116), 1503)]
```

Tiếp đến em sẽ chọn N-grams tốt nhất dựa vào hướng dẫn ở phần “j) Chọn N-grams tốt nhất” như sau:

```
from os import listdir
from os.path import isfile, join
directories = ["/content/drive/MyDrive/(Lab2)Dataset/Benign PE Samples", "/content/drive/MyDrive/(Lab2)Dataset/Malicious PE Samples"]
N = 2

Ngram_counts_all_files = collections.Counter([])
for dataset_path in directories:
 all_samples = [f for f in listdir(dataset_path) if
isfile(join(dataset_path, f))]
 for sample in all_samples:
  file_path = join(dataset_path, sample)
  Ngram_counts_all_files += binary_file_to_Ngram_counts(file_path, N)

K1 = 1000
K1_most_frequent_Ngrams = Ngram_counts_all_files.most_common(K1)
K1_most_frequent_Ngrams_list = [x[0] for x in K1_most_frequent_Ngrams]

def featurize_sample(sample, K1_most_frequent_Ngrams_list):
 """Takes a sample and produces a feature vector. The features are the counts of the K1 N-grams we've selected."""
 K1 = len(K1_most_frequent_Ngrams_list)
 feature_vector = K1 * [0]
 file_Ngrams = binary_file_to_Ngram_counts(sample, N)
 for i in range(K1):
  feature_vector[i] = file_Ngrams[K1_most_frequent_Ngrams_list[i]]
 return feature_vector
```

```
directories_with_labels = [("/content/drive/MyDrive/(Lab2)Dataset/Benign PE Samples", 0), ("/content/drive/MyDrive/(Lab2)Dataset/Malicious PE Samples", 1)]
X = []
y = []
for dataset_path, label in directories_with_labels:
 all_samples = [f for f in listdir(dataset_path) if isfile(join(dataset_path, f))]
 for sample in all_samples:
  file_path = join(dataset_path, sample)
  X.append(featurize_sample(file_path,
K1_most_frequent_Ngrams_list))
  y.append(label)

from sklearn.feature_selection import SelectKBest, mutual_info_classif, chi2
K2 = 10

import numpy as np
# Chọn N-grams phổ biến
X = np.array(X)
X_top_K2_freq = X[:,:K2]
print("N-grams phổ biến: ", X_top_K2_freq)

# Chọn N-grams có xếp hạng cao theo thuật toán mutual information
mi_selector = SelectKBest(mutual_info_classif, k=K2)
X_top_K2_mi = mi_selector.fit_transform(X, y)
print("N-grams có xếp hạng cao theo thuật toán mutual information: ", X_top_K2_mi)

# Chọn N-grams có xếp hạng cao theo thuật toán chi squared
chi2_selector = SelectKBest(chi2, k=K2)
X_top_K2_ch2 = chi2_selector.fit_transform(X, y)
print("N-grams có xếp hạng cao theo thuật toán chi squared: ", X_top_K2_ch2)
```

Đoạn code trên có thể hiểu sơ như sau, code đi qua hai thư mục chứa tệp benign và malicious, tính toán n-grams (trong trường hợp này là bigrams, với N=2) cho mỗi tệp nhị phân trong thư mục.

Sau khi tính toán n-grams cho tất cả các tệp, code lấy ra 1000 n-grams phổ biến nhất để sử dụng làm đặc trưng cho việc huấn luyện mô hình phân loại.

Code tiếp tục bằng cách tạo ra vector đặc trưng cho mỗi tệp nhị phân bằng cách đếm số lượng xuất hiện của 1000 n-grams phổ biến nhất trong từng tệp. Mỗi vector đặc trưng này tương ứng với một hàng trong ma trận đặc trưng X, và nhãn tương ứng (0 cho benign, 1 cho malicious) được lưu trong vector y.

Code sử dụng ma trận đặc trưng X và vector nhãn y để áp dụng ba phương pháp chọn đặc trưng khác nhau là chọn N-grams phổ biến, chọn N-grams có xếp hạng cao theo thuật toán mutual information, chọn N-grams có xếp hạng cao theo thuật toán chi squared.

Kết quả của đoạn code trên như sau:

```
N-grams phổ biến:  [[  10935    4673       7     610      26       0      13     565     248
      14]
 [  15237    2604     866    1848    1332    1343     630      22      17
     630]
 [   4963     282      88     129     222     573     120       6       0
      67]
 [  36882    1921    6191    6527    2784       0    1435     413    2770
    1319]
 [ 140825   11831       7    1350      25       2       3     527     670
       8]
 [   7159    2219       3    1136      29       0       0     350     949
       5]
 [  18655    1518       3    1176     173       1       1     265    1621
       5]
 [  17320    1851      34     770      43       1       3     295     402
      21]
 [  71210    5507       2     259      10       1       1     210     225
       2]
 [  15222     699       0     186       9       1       0     202     101
       2]
 [  19249     788    2849    2681    2903       1    1366     319     548
     531]
 [  10478     469     732     338     479       0     303     131     485
     377]
 [   7719     274     628     384     376       0     185      61     458
     274]
 [  29162    3327    4117    5514    3085       0    2559     727    1124
    1164]
 [  12594     254     449     578     462       0     280      71     561
     101]
 [  52163    8162   15778    6626    7701       0    5849    2012    5038
    4520]
 [  64674    6729   16401    6441    7972       4    5474    1828    5710
    4224]
 [  10779     458    1023     519     535       0     300     102     677
     434]
 [   8958     426     759     267     523       0     358     123     382
     356]
 [   9772     437     875     450     405       0     300      97     536
```

```
-
     362]
[  11322     717    1665    1721    1049       0     716     119    1229
     784]
[ 427928   86437  108028   57985   37707      23   27497   24224   91987
   43216]
[  11537     711    1233     497     756       0     498     182     900
     467]
[   7969     230     543     125     436       0     251      50     173
     138]
[   7670     171     590     263     449       0     354      84     374
     225]
[  13045     743    2091    2338    1492       0     740     310     464
     389]
[  21528    2305    2779    4933    1921       0    1815     445     837
    1003]
[   4386      42      80      67      86       0      39      10     116
      47]
[   8054     117     594     336     376       0     153      70     987
     249]
[  30694    4419    2415    4993    2102       9    1124    1894    1807
     674]
[   3628     175       0      94       2       0       0      69      84
       0]
[   4160     494       0      99      11       0       0     115     137
       0]
[  17578    2488       3     421      10      15       3     598     539
       1]
[  97548   28401      29    2656      64       2      38    3818    3167
      35]
[   3845     157       0      63       1       0       0      41     146
       0]
[   5507     471       1     121      12       0       1      69      97
       0]
[  58780    2185       0     220      18      73       1     215     219
       1]
[   4993     774       6     157      22       0       0      93     118
       1]
[  31971    3213       5    1096      52       1       3     725     935
       1]
[  39670    5317       9    1337      38       2       3    1331    1085
      11]
[  49506    9704       6    1574      66       0       5    1108    1111
       3]
[  29685    4199      14    1567      73       0       5    1561    1118
      29]
```

```
[  80989   11591      10    2227     177      15       2    3036    1549
      19]
[  20592    5867      17    1994      74       1       5    1025    4344
      28]
[   5346     848       3     490       6       0       0     247     437
      10]
[   3096      47       0      37       2       0       0      19     108
       0]
[  14969    1799       1     261      18       1       5     289     100
       1]
[  11140     311       0     169       6       6       3     148      97
       4]
[  14685     795       4     826      54       0       2     427     699
       4]
[   3119      78       0      56       2       0       0      17     100
       0]
[   4216     114       0      57       6       0       0      87     137
       0]
[   4990     524       2     107       7       0       1      96     533
       0]
[  29271    6270       6    2642      73       1       1    2177     835
      10]
[   2481       1       0     133       9       1       1       3       0
       0]
[   3640      10       0     237       8       1       2       4       0
       0]
[   9410    4453       1     318      37       2       0     421     481
       1]
[  65038   11212      15    2845      94      10       5    1891    2811
      57]
[ 110759  130086      10    1925      72       6       2    5472    1490
       5]
[  20138    2279       2    1150      30       0       1     357    1549
       0]
[  17061    1713       1     322       9       2       1     218     503
       0]
[      9       0       1       3       1       0       2       2       0
       0]
[      9       1       1       5       2       1       1       2       2
       3]
[      8       1       0       1       1       0       2       2       1
       1]
[      9       0       0       1       1       1       0       0       0
       0]
[     12      11       1       5       0       3       1       3       1
       1]
```

```
[    10     1     0     3     0     0     0     0     1
     0]
[     5     0     1     3     1     0     1     3     1
     1]
[     8     1     0     1     0     0     0     0     1
     0]
[     8     0     0     1     0     0     0     0     0
     0]
[     6     1     1     3     0     0     1     1     0
     0]
[     7     3     0     5     0     0     0     0     0
     0]
[     6     4     0     1     0     3     0     0     0
     0]
[     8     3     0     4     0     1     0     1     0
     0]
[     6     0     0     2     1     2     0     0     3
     0]
[     9     1     0     2     1     0     0     0     0
     0]
[    11     1     0     1     0     0     0     0     1
     0]
[     5     0     1     3     0     1     1     1     0
     0]
[  44667  6048  9250  5880  4710     1  2915  1507  3073
  2474]
[   5359    91   257   156   247     0   124    39   139
   116]
[  35697  6257  6263  7671  2239  6059  1337  1591    28
  2724]
[2028238 631237 216934 118021 103955 166298 111696 82528    87
 73376]
[  14677  1493  1949  1636   920  1459   679   308    18
   799]
[ 802224 448826 22478 17957  7827 11493  6227 25273    42
  7087]
[  20146  4014    13  1342    60     0     3  1062  2495
     9]
[  21401  1040  4948  3695  2918     0  1876   352  1554
  1304]
[  11257   827  1223   463   681     0   507   181   440
   456]
[  65606 24303    65  6419    80  8770    25   367     2
     25]]
```

```
N-grams có xếp hạng cao theo thuật toán mutual information:  [[  10    2    2    3   12   35    2   14    0    0]
 [   0    1    8   13   21   13   10    8  140    0]
 [   0    0    0    2    1    0    0    1    1    0]
 [   5   13   36   17  144   32    2   48   92    0]
 [ 162    8    3    5    4   23   15   51    2  736]
 [   1    5    0   17    5   17    2   11    0    0]
 [   0    3    0   11    2   28    2    7    1    0]
 [  38    2    2    4    2   13    2   17    1    0]
 [  81    4    1    0    3    3    7    8    1  368]
 [   7    0    2    2    1    2    5    4    1    8]
 [   2    7   26   11   29   17    8   21   49    0]
 [   5    4    2    1    8    6    3    2    3    1]
 [   0    3    2    0   10    5    2    1    7    0]
 [   3   24   45   35   65   43    6   37   69    2]
 [   0    0    4    2   14    1   19    2    3    0]
 [   6  113  210  129  218  179    3   93  227    5]
 [  16   76  239  122  228  225   24   53  933   52]
 [   0    5    3    2   10    6    6    1   18    0]
 [   0    4   16    0    8   16    0    4    6    0]
 [   1    4    8    2    2    2   11    3   10    1]
 [  12    8   16   20   14   40   11    7   15    0]
 [  18  552  515  217  396  488   46  319  157   55]
 [   0    9   23    8   49   21   15    9   18    0]
 [   1    1    1    0    8   12   11    2    1    0]
 [   1    2    6    1    9    2    8    4    7    0]
 [   1    5   18    4    9   11    3   16   42    0]
 [   3   19   11    6  105   25   23   30   40    0]
 [   0    0    1    0    2    2    0    0    0    0]
 [   0    2    6    2    7    6   12    1    5    0]
 [  12   18   21    3   29   43   23   28   25    8]
 [   0    0    0    0    1    0    0    0    0    0]
 [   0    0    0    6    0    9   11    2    0    0]
 [   8    2    2    3    1   12    4    4    1   59]
 [  16   11    7   37    5   99   36  105    9  205]
 [   0    1    0    1    0    2    0    0    0    0]
 [   0    0    0    0    0    5   24    0    0    0]
 [ 141    0    1    0    1    0   10    5    0  239]
 [   0    0    0    5    0   21    0    2    0    0]
 [  10    1    0    9    2   17    5   22    2    0]
 [  29    2    0    1    0   39   18   46    2   29]
 [ 492    2    5   35    0   35   13   34    1    1]
 [  27    3    0   20    0   38   22   67    5   45]
 [  57    2    6   19    0   45   68   18    4  154]
 [   5    2    0   54    2   39    8   79    4    0]
 [   1    0    1    1    3   18    2    0    1    0]
 [   0    0    0    0    0    1    0    0    0    0]
 [  14    1    0    4    2   55    8    6    3   10]
```

```
[   6    0    0    4    1    3    7    5    4   13]
[   0    0    0    5    1   89    3    4    1    0]
[   0    0    0    0    0    0    0    0    0    0]
[   0    0    0    0    0    1    0    3    0    0]
[   1    1    0    3    0    5    1    7    0    0]
[  12    2    0   34    8  110   14   40    5   10]
[   0    0    0    0    0    0    2    8    0    1]
[   0    0    0    0    3    2    3    4    0    1]
[   2    1    0    7    0   11    5    8    0    0]
[  61    4    0   54    0  100   37   90    9  143]
[   8    2    1    5    0   19   22   30    2   61]
[  12    2    2    5    0   14   12   30    1   30]
[  12    1    0    1    0    5    8   11    0   30]
[   2    1    2    0    0    0    2    3    2    2]
[   2    2    0    2    0    2    0    1    0    0]
[   0    0    2    0    1    1    1    0    1    1]
[   0    0    0    0    1    0    0    0    0    0]
[   2    2    1    1    3    2    1    0    1    5]
[   0    1    0    3    1    0    0    0    2    0]
[   1    1    0    0    1    0    1    1    1    1]
[   0    0    0    1    0    0    0    0    0    0]
[   0    0    0    0    0    0    0    0    0    0]
[   2    1    1    0    0    0    2    0    1    0]
[   2    0    0    0    0    0    0    0    0    0]
[   3    0    1    0    0    0    0    1    1    1]
[   0    0    0    0    0    0    0    0    0    0]
[   0    0    0    0    1    1    0    1    0    0]
[   0    0    0    1    1    0    1    0    0    1]
[   0    1    0    1    0    0    0    0    0    0]
[   1    0    1    1    0    0    2    0    0    1]
[  15   75   95  211  149   62    9   47   87    2]
[   0    2    2    0    6    3    0    1    4    0]
[   5   36   42   63   15   10   46   25   10    3]
[1461 1545 1229 1049  958  410 1805  900  646  320]
[   1   10   14    9   11    7   20   15    6    1]
[  12  168  121  152  120   24  234  173   54  180]
[   2    0    0   22    0   58    5   33    3    0]
[   2   20   35   18   63   43   10   24   17    0]
[   4    9    9   10   18   12   14    4    6    0]
[   9   13    6  314   12   57   22  133   20    4]]
```

```
N-grams có xếp hạng cao theo thuật toán chi squared:  [[  615   433   574   335    46   552    84     4     7     1]
 [  171    32   125     6    11    46     1     4     0     9]
 [   19     0    10     1     3     6     0     0     0     6]
 [  436   447    80     8   623    12     1     6     0     0]
 [   75   192   165    72    57    54    19     3    27     2]
 [   62   191   218   104    57    61    11     4    11     0]
 [  149   222   147    90    54   121    40     3    24     0]
 [   72   127   126    48    30    81    27     5    14     1]
 [   16    22    42     9     5    13     5     1     7     1]
 [    9    26    26     7     3    14     1     0     1     1]
 [  130   115    78     6    71    71     1     0     1     0]
 [  131    10    19     1     8     2     2     3     0     0]
 [   90     6    12     1     7     2     1     1     0     0]
 [  294    69    74    54    37     6     7    10     1     0]
 [   41    13    14     4     1    17     1     0     0     0]
 [ 1072   266   430    68   142    50     4    18     0     0]
 [ 1038   376   391   134   189   113     9    24     3     1]
 [  140    10    17     1     9     4     1     1     0     0]
 [   92    22    19     2     5     0     1     0     0     0]
 [  105    14    11     3     3     2     1     0     0     0]
 [  280    23    26    12     3    11     1     0     0     0]
 [14069   916   961   522   441   400    41    84    48     6]
 [  168     7    34     9     7    13     1     1     1     0]
 [   45    42    35     4    18     1     0     0     0     0]
 [   66    12    11     4    16     0     1     0     0     0]
 [  101   105    85     7    53    41     1     0     1     0]
 [  340   127    95    13    38     8     2     2     0     0]
 [   15     2     2     0     0     0     1     0     0     0]
 [   70     4    16     1    10     1     0     0     0     0]
 [  217    44   148    10    19    23     2     1     0     1]
 [    9    19    13     2     0     5     4     2     0     0]
 [   66    77    61    10    12    31    20     0     3     0]
 [   36   143   100    35    21    31    10     2    21     0]
 [  164  1087   719   319   217   174    32   256    77     4]
 [   10    15    22    14     0     3     4     0     0     0]
 [   88    36    84   103    10    25    31     1     3     0]
 [    5    18    18     3     1    16     3     2     0     2]
 [   95   103    91    41    51    60    17     1     5     0]
 [   79   221   165    98    43    47    26     4    19     0]
 [  110   471   284   126    77    98    22     3    28     0]
 [  224   457   422   265    70   108    90     3    47     0]
 [  205   392   312   300   110   135    74    30    32     2]
 [  206   284   423   171    43   114    97     5    34     0]
 [  328   776   626   485   188   257    84    16    59     0]
 [  271   128   242    44    81   123    92     1    10     0]
 [    5    15    11     2     0     3     2     0     0     0]
```

```
[  120    37    69    37    12   103    17     5     8     0]
[    8    17    22     5     1    11     8     1     0     0]
[  135   230   228    20    50    40    12     0     8     0]
[    6    16    11     3     0     3     4     0     0     0]
[    7    53    17     8     8     4     2     2     3     0]
[   30    76    45    26    12    18    10     0    13     0]
[  377   489   543   129   112   179    53     1    17     0]
[    0     0     6     0     0     1     1     0     0     0]
[    0     0     5     0     0     1     0     0     0     0]
[   81    76   140    34    18    31    43     1     8     0]
[  356   587   734   396   149   300   103    11   111     0]
[  104   193   241    87    52    66    33     3    24     1]
[   75   147   128    65    32    51    31     3    26     0]
[   40    75    75    29     9    25    14     0    12     0]
[    3     0     3     1     2     0     0     2     1     4]
[    2     1     2     5     2     1     1     4     0     2]
[    0     0     1     3     0     1     1     0     1     2]
[    0     0     0     0     1     0     0     0     0     0]
[    2     2     0     0     1     1     1     1     4     3]
[    0     0     1     0     1     0     0     0     0     0]
[    1     1     1     2     2     0     0     0     2     1]
[    0     0     1     0     1     0     0     0     0     0]
[    0     0     0     0     0     0     0     0     0     0]
[    0     0     0     1     1     0     1     1     2     2]
[    1     0     0     0     1     1     0     0     0     1]
[    1     1     0     0     1     0     1     1     2     1]
[    1     0     0     0     0     2     0     0     0     1]
[    3     1     1     3     1     1     0     0     2     1]
[    0     1     0     0     2     0     0     0     0     0]
[    0     0     0     0     1     0     0     0     0     0]
[    1     1     0     0     3     4     0     0     1     0]
[  699   173   255    75    96    33     4    11     0     0]
[   30     2    11     0     0     4     0     0     0     0]
[  722    35    86    39    16     8     4     1     1     0]
[21020  3204  3036  1540  1109  1075    59    71    36     1]
[  220    23    33     4    11    16     2     1     0     0]
[ 1971   436   367   148   143   376     4    13     0     0]
[   68   523   248   124    81   111    20     4    33     0]
[  391    98   130    31    55    32     6     6     0     0]
[  137    19    26     2    15     2     3     2     0     0]
[ 8040  7005  5341  3914  3485  2884  6445  3096  2914  2928]]
```

Câu 8: Sinh viên hoàn thành các bước

Bước 1: Tạo list các mẫu và gán nhãn cho chúng:

```
import os

directories_and_labels = [("/content/Benign PE Samples/", 0),
                          ("/content/Malicious PE Samples/", 1)]

list_of_samples = []
labels = []
N_spec = 2 # N-grams
```

```
[ ] for ds_path, label in directories_and_labels:
      samples = [item for item in os.listdir(ds_path)]
      for sample in samples:
        file_path = os.path.join(ds_path, sample)
        list_of_samples.append(file_path)
        labels.append(label)
```

```
[ ] # Xem 5 samples và nhãn đầu tiên
    list_of_samples[:5], labels[:5]
```

```
(['/content/Benign PE Samples/IMEWDBLD.EXE',
  '/content/Benign PE Samples/InspectVhdDialog6.2.exe',
  '/content/Benign PE Samples/imjpuexc.exe',
  '/content/Benign PE Samples/logoff.exe',
  '/content/Benign PE Samples/isoburn.exe'],
 [0, 0, 0, 0, 0])
```

```
[ ] # Xem 5 samples và nhãn cuối cùng
    list_of_samples[(len(list_of_samples) - 5):], labels[(len(labels) - 5):]
```

```
(['/content/Benign PE Samples/LicensingUI.exe',
  '/content/Benign PE Samples/IMEPADSV.EXE',
  '/content/Benign PE Samples/IMJPDCT.EXE',
  '/content/Benign PE Samples/lpksetup.exe',
  '/content/Malicious PE Samples/aapt.exe'],
 [0, 0, 0, 0, 1])
```

Bước 2: Chia dữ liệu train và test:

```
from sklearn.model_selection import train_test_split
```

```
[ ] samples_train, samples_test, target_train, target_test = train_test_split(
        list_of_samples,
        labels,
        test_size=0.3,
        stratify=labels,
        random_state=42
    )
```

Bước 3: Các hàm lấy thuộc tính:

```
[ ] import collections
    from nltk import ngrams
    import numpy
    import pefile
```

```
[ ] def read_file(file_path):
      with open(file_path, "rb") as bin_file:
        data = bin_file.read()
        return data
```

```
[ ] def byte_seq_to_Ngrams(byte_seq, N_par):
      Ngrams_par = ngrams(byte_seq, N_par)
      return list(Ngrams_par)
```

```
[ ] def bin_file_to_Ngrams_count(file_path, N_par):
      file_seq = read_file(file_path)
      file_Ngrams = byte_seq_to_Ngrams(file_seq, N_par)
      return collections.Counter(file_Ngrams)
```

```
[ ] def get_Ngrams_features_from_samples(sample, K1_most_freq_Ngrams_list):
      K1 = len(K1_most_freq_Ngrams_list)
      feature_vector = K1 * [0]
      file_Ngrams = bin_file_to_Ngrams_count(sample, N_spec)
      for i in range(K1):
        feature_vector[i] = file_Ngrams[K1_most_freq_Ngrams_list[i]]
      return feature_vector
```

```
[ ] def preprocess_imports(list_of_DLLs):
      """ Normalize the name of the imports of a PE file. """
      temp = [x.decode().split(".")[0].lower() for x in list_of_DLLs] # View the transforming of below example
      return " ".join(temp)
```

```
def get_imports(pe):
  """ Get a list of the imports of a PE file """
  list_of_imports = []
  for entry in pe.DIRECTORY_ENTRY_IMPORT:
    list_of_imports.append(entry.dll)
  return preprocess_imports(list_of_imports)
```

```
[ ] def get_section_names(pe):
      """ Get a list of the section names of a PE file """
      list_of_sections = []
      for sect in pe.sections:
        normalized_name = sect.Name.decode().replace("\x00", "").lower()
        list_of_sections.append(normalized_name)
      return "".join(list_of_sections)
```

Bước 4: Chọn 100 thuộc tính phổ biến với 2-grams:

```
[ ] Ngrams_count_all = collections.Counter([])
    for sample in samples_train:
      Ngrams_count_all += bin_file_to_Ngrams_count(sample, N_spec)
    K1 = 100
    K1_most_common_Ngrams = Ngrams_count_all.most_common(K1)
    K1_most_common_Ngrams_list = [x[0] for x in K1_most_common_Ngrams]
```

Bước 5: Trích xuất số lượng N-grams count, section names, imports và số lượng sections của mỗi mẫu trong train-test:

```
imports_corpus_train = []
num_sect_train = []
sect_name_train = []
Ngram_feat_list_train = []

y_train = []
```

```
[ ] for i in range(len(samples_train)):
      sample = samples_train[i]
      try:
        # Get all required parameters with predefined functions
        Ngram_features = get_Ngrams_features_from_samples(sample, K1_most_common_Ngrams_list)
        pe = pefile.PE(sample)
        imports = get_imports(pe)
        n_sections = len(pe.sections)
        sec_names = get_section_names(pe)

        # Put above value into lists
        imports_corpus_train.append(imports)
        num_sect_train.append(n_sections)
        sect_name_train.append(sec_names)
        Ngram_feat_list_train.append(Ngram_features)

        # Target train
        y_train.append(target_train[i])
      except Exception as e:
        print(sample + ":")
        print(e)
```

Kết quả:

```
[ ] /content/Benign PE Samples/LogCollector.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/lpq.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/inetinfo.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/InspectVhdDialog.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/hvsirdpclient.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/InetMgr6.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/LogCollector.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/iissetup.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/inetinfo.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/InetMgr.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/hvsirpcd.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/iisreset.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/lpr.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/InspectVhdDialog6.3.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/iisrstas.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/hvsirdpclient.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/ldifde.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/InetMgr.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/InspectVhdDialog6.2.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/lpr.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/LxRun.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/iisreset.exe:
    'DOS Header magic not found.'
    /content/Benign PE Samples/LxRun.exe:
    'DOS Header magic not found.'
```

```
/content/Benign PE Samples/iisreset.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/LxRun.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/InetMgr.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/iisrstas.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/lpq.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/iissetup.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/ldp.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/InspectVhdDialog.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/hvsirpcd.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/LogCollector.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/lpq.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/InspectVhdDialog.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/lpr.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/ldp.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/iissetup.exe:
'DOS Header magic not found.'
```

Bước 6: Sử dụng hàm băm tfidf để chuyển imports, section names từ văn bản thành dạng số:

```
from sklearn.feature_extraction.text import HashingVectorizer, TfidfTransformer
from sklearn.pipeline import Pipeline
```

```
[ ] imports_featurizer = Pipeline(
        [
            ("vect", HashingVectorizer(input = "content", ngram_range=(1,2))),
            ("tfidf", TfidfTransformer(use_idf = True,)),
        ]
    )
```

```
[ ] sect_name_featurizer = Pipeline(
        [
            ("vect", HashingVectorizer(input = "content", ngram_range= (1,2))),
            ("tfidf", TfidfTransformer(use_idf = True))
        ]
    )
```

```
[ ] imports_corpus_train_transformed = imports_featurizer.fit_transform(imports_corpus_train)
```

```
[ ] sect_name_train_transformed = sect_name_featurizer.fit_transform(sect_name_train)
```

Bước 7: Kết hợp các vector thuộc tính thành 1 mảng:

```
from scipy.sparse import hstack, csr_matrix
```

```
X_train = hstack(
    [
        Ngram_feat_list_train,
        imports_corpus_train_transformed,
        sect_name_train_transformed,
        csr_matrix(num_sect_train).transpose(),
    ]
)
```

Bước 8: Huấn luyên bằng phân loại Random Forest cho tập train:

```
from sklearn.ensemble import RandomForestClassifier
```

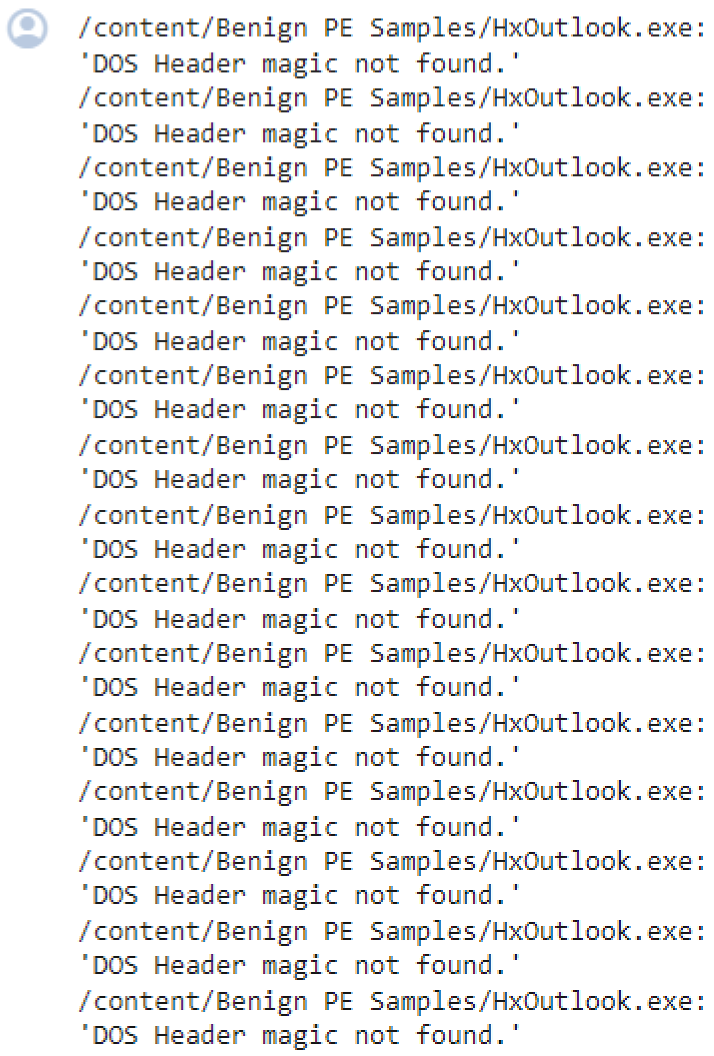
```
[ ] clf = RandomForestClassifier(n_estimators = 100)
    clf = clf.fit(X_train, y_train)
```

Bước 9: Thu thập các thuộc tính của tập test, giống như tập huấn luyện:

```
[ ] import_corpus_test = []
    num_sect_test = []
    sect_names_test = []
    Ngram_feat_list_test = []

    y_test = []
```

```
[ ] for i in range(len(samples_test)):
      test = samples_test[i]
      try:
        # Get all required parameters with predefined functions
        # The input when getting N-grams features is still "sample"
        Ngram_features = get_Ngrams_features_from_samples(sample, K1_most_common_Ngrams_list)
        pe = pefile.PE(test) # Get test PE file
        imports = get_imports(pe)
        n_sections = len(pe.sections)
        sec_names = get_section_names(pe)

        # Put above value into lists
        import_corpus_test.append(imports)
        num_sect_test.append(n_sections)
        sect_names_test.append(sec_names)
        Ngram_feat_list_test.append(Ngram_features)

        # Target train
        y_test.append(target_test[i])
      except Exception as e:
        print(sample + ":")
        print(e)
```

Kết quả:



```
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
/content/Benign PE Samples/HxOutlook.exe:
'DOS Header magic not found.'
```

Bước 10: Chuyển đổi vector từ thuộc tính test, và kiểm tra kết quả của trình phân loại:

```
[ ] import_corpus_test_transformed = imports_featurizer.transform(import_corpus_test)
```

```
[ ] sect_names_test_transformed = imports_featurizer.transform(sect_names_test)
```

```
[ ] X_test = hstack(
        [
            Ngram_feat_list_test,
            import_corpus_test_transformed,
            sect_names_test_transformed,
            csr_matrix(num_sect_test).transpose()
        ]
    )
```

```
[ ] print("The score of our classifier is as follow: ")
    print(clf.score(X_test, y_test))
```

```
The score of our classifier is as follow:
0.984375
```