Actividad 2 de evaluación continua a distancia para la asignatura: "Introducción al Aprendizaje Automático para Ciberseguridad"

curso 2019-20

1. Información previa

Es importante que en primer lugar **lea todo este documento completo detenidamente** y hasta el final, ya que proporciona paso a paso toda la información necesaria para realizar la actividad. También es necesario haber leído previamente el Manual Didáctico en el curso virtual de la asignatura y haber seguido las recomendaciones de estudio incluidas en el mismo.

Aunque algunas de las instrucciones de preparación de datos y requisitos coinciden con las dadas para la actividad 1, en este documento se incluirán completas para que se pueda realizar de forma independiente, pero es conveniente repasarlas para ver las diferencias y los posibles requisitos adicionales.

1.1. Resumen

Se propone la realización de una práctica guiada sencilla basada en el código de ejemplo proporcionado con el libro base que se indica en el Manual Didáctico. En este caso se trata de una práctica sobre el tema 3 de clasificación, siguiendo el ejemplo dado en el libro base (capítulo 2) utilizando los ficheros de respuesta a peticiones específicas por HTTP en una *BotNet* con servidores de control conocidos, y también a otras direcciones normales, para entrenar un clasificador que los reconozca.

1.2. Forma de entrega

Esta es un tarea individual y personal de cada estudiante, no se permiten grupos. El resultado de la actividad debe redactarse en un documento, cuyo contenido y estructura se especifica detalladamente más adelante (sección 1.4), convertido a formato **PDF**. El documento se enviará únicamente a través de la tarea correspondiente **dentro del curso virtual** (en la página de "Actividades Evaluables" del menú izquierdo): `https://2020.cursosvirtuales.uned.es/dotlrn/posgrados/asignaturas/31109097-20/?page_num=3`

Por favor, **no** envíen la respuesta por ningún otro medio, ni por correo electrónico, para evitar confusiones, extravíos y problemas de gestión. Tampoco se aceptarán entregas fuera del plazo establecido para todos los estudiantes.

En caso de que desee añadir algún otro fichero muy relevante en su respuesta (código fuente, hoja de cálculo, etc.), debe empaquetar todos los ficheros juntos, incluido el documento PDF obligatorio, en un archivo comprimido compatible ZIP o RAR y subir ese archivo comprimido (en lugar del documento) a la tarea correspondiente dentro del curso virtual.

1.3. Plazo de entrega

La entrega se debe realizar subiendo el fichero correspondiente en la tarea del curso virtual **antes del día 21 de junio de 2020 a las 23:55h.** Se recomienda no esperar hasta el último día, para evitar las saturaciones[1] o problemas puntuales de acceso a internet, etc. Si surge algún problema para el envío, realice lo antes posible una consulta directamente al coordinador del equipo docente por email (pero no envíe el fichero por ese medio) a <jras@dia.uned.es>.

[1] Téngase en cuenta que entre los días **25 de mayo al 13 de junio** en aLF (la plataforma de cursos virtuales), y entre los días **15 al 19 de junio** en AvEx (plataforma `unedenlinea.es`), se realizarán de **9h a 20h** pruebas sincronizadas en otras asignaturas, que podrían causar congestiones puntuales. Durante esos días se recomienda subir los ficheros fuera de ese horario (es decir, hacerlo preferiblemente después de las 20h y antes de las 8h.).

La respuesta de la actividad se puede enviar en el curso virtual tantas veces como se quiera hasta el plazo indicado, ya que solamente la **última versión** se guarda y es la única que se tendrá en cuenta para la evaluación al finalizar el plazo.

Se recomienda verificar que el fichero guardado en el servidor es el correcto, descargándolo después del envío para comprobar que es la versión correcta.

1.4. Contenidos y estructura del documento de respuesta

El documento de respuesta puede redactarse en cualquier editor, pero se debe **exportar a PDF** para enviarlo. Por ejemplo, se puede usar el editor de LibreOffice que permite exportar a PDF, es gratuito y está disponible para todas las plataformas (ver `http://libreoffice.org/` para obtenerlo). También es muy recomendable el programa LyX (ver `http://lyx.org/`) que puede generar PDF de calidad fácilmente y está disponible en la mayoría de distribuciones de Linux, y también para MS-Windows o Mac.

En el documento de respuesta (indicado en el apartado 1.2) se deben especificar los resultados de la actividad, y también las explicaciones de cómo se han obtenido éstos, ordenados según el siguiente **esquema**:

1. Datos del estudiante: Nombre, Apellidos, DNI o pasaporte y email de contacto. También se puede incluir opcionalmente un número de teléfono de contacto.

2. Información sobre el entorno y programas usados para esta actividad (incluyendo números de versiones): Sistema operativo, entorno de escritorio, versión de Python y editor de texto para generar el documento PDF. Adicionalmente se puede incluir información de otros programas (versiones de paquetes, etc.) o medios utilizados para la realización.

3. Breve resumen de la actividad a realizar.

4. Respuestas de cada uno de los apartados pedidos en la actividad (ver apartado 3). Es necesario separar y etiquetar claramente cada apartado y, aparte de la información o datos específicos requeridos, se recomienda incluir lo siguiente:

   a) Qué valores de parámetros se han usado para obtener la respuesta.
   b) Qué modificaciones de código se han hecho respecto a los originales.
   c) Qué fallos o errores se han encontrado y cómo se han solucionado.

5. Comentarios y opiniones: Dificultades o problemas encontrados, programas o ayudas utilizadas, comentarios sobre la realización de la actividad, etc.

6. Bibliografía o fuentes de información: Documentos, páginas web, libros, etc. consultados para la realización (con datos de título y editorial o URL de localización, etc.).

Es muy importante para la evaluación, seguir el esquema dado e **incluir explicaciones** de cómo se han realizado los cálculos (qué lista de datos, operación, resultado, etc.) para que se pueda calificar correctamente.

Utilice un formato sencillo y claro, con una redacción estructurada en apartados numerados en el documento. Por favor, no utilice imágenes ni formatos decorativos innecesarios, pues pueden valorarse negativamente.

Para incluir datos y resultados obtenidos en comandos de consola de texto, se debe copiar el texto y "pegarlo" como texto en el documento. Se le puede dar formato con un tipo de letra de paso fijo (*fixed font*) y preservar los saltos de línea[2]. No se deben insertar capturas gráficas de la pantalla de las ventanas de terminal de consola que ocupan bastante más y pueden tener peor calidad.

2. Instrucciones previas

Para la realización de la actividad, es necesario haber instalado y probado el código de ejemplo que se proporciona con el libro base indicado en el Manual Didáctico en el curso virtual de la asignatura. A continuación se indican los requisitos y pasos previos para conseguir que dicho código funcione adecuadamente.

[2]En el procesador LyX se puede insertar el texto dentro de un "Listado de Código" que preserva el espaciado.

2.1. Requisitos

2.1.1. Sistema operativo y paquetes necesarios

Los programas necesarios en esta actividad utilizan el código proporcionado con el libro base en el repositorio de GitHub correspondiente, por tanto es muy recomendable utilizar una sistema operativo Linux, preferiblemente una distribución derivada de Debian o Ubuntu reciente.

Los cálculos para la actividad se pueden realizar dentro de una máquina virtual con al menos 2GiB de memoria. Se pueden utilizar imágenes de máquinas virtuales (VDI o virtual disk image) ya preinstaladas de Debian, Ubuntu, Linux-Lite o cualquier otra desde osboxes.org. También hay muchas imágenes preinstaladas en virtualboxes.org y en descargarmaquinasvirtuales.com (en español). Las imágenes de disco de VirtualBox o de VMWare se pueden convertir también para usar en Parallels de Mac.

Para algunos comandos adicionales usados, será necesario tener instalados (con "apt" o la herramienta de paquetes propia de la distribución de Linux utilizada) los paquetes siguientes: `wget`, `gzip`, `grep`.

En este caso, además será necesario tener instalado algún paquete que proporcione **el comando** `lzma`, u otro equivalente o compatible, para poder descomprimir los ficheros de ese tipo, como por ejemplo alguno de los paquetes Debian o Ubuntu: `lzma`, `zx-utils`, `p7zip`, etc. Además, también será necesario el paquete: `graphviz`, para exportar a formato PNG la descripción en formato .dot del árbol de decisión generado.

Para la visualización de la proyección de datos (con Matplotlib), será necesario un entorno gráfico compatible X11, para lo cual sirve cualquier entorno de escritorio de los habituales en Linux (Gnome, KDE, XFCE, LXDE, etc.).

2.1.2. Entorno Python y módulos

Aunque el código de ejemplo utilizado en esta actividad está escrito originalmente en Python2, se recomienda utilizar Python3 para su realización, ya que las modificaciones necesarias para ello son triviales (tal como se indica al final del apartado 2.2.2).

Siempre que sea posible, se recomienda instalar los paquetes y bibliotecas de Python del sistema (con "apt" en sistemas Debian o Ubuntu, o bien la herramienta equivalente de la distribución Linux usada), e instalar únicamente con "pip3" aquellos módulos que no existan como paquetes del sistema.

Para esta actividad será necesario tener instalados los paquetes: `python3-gevent` y `python3-ssdeep`, aparte de los paquetes ya utilizados en la actividad 1: `python3-sklearn`, `python3-matplotlib`, `python3-tk`, `python3-h5py`, `python3-requests`, `python3-dev`, `2to3`, con lo que se instalarán adicionalmente como dependencias otros paquetes necesarios como: `python3`, `python3-numpy`, `python3-scipy`, etc.

2.2. Ficheros necesarios

2.2.1. Código y datos de ejemplo

Los ficheros de código fuente necesarios se pueden descargar directamente del repositorio público GitHub[3] en un archivo comprimido (64.2MB):
`https://github.com/cylance/IntroductionToMachineLearningForSecurityPros/archive/master.zip`
que, una vez descomprimido (234.7MB), contiene el directorio:
`IntroductionToMachineLearningForSecurityPros-master/IDPanel/` con los ficheros que se utilizarán en esta actividad. En ese directorio hay 3 ficheros (`prevectors.json.lzma`, `raw_feature_vectors.json.lzma` y `raw_features.json.lzma`) comprimidos con formato lzma, que es necesario descomprimir antes de poder usarlos en esta actividad. Para ello, simplemente hay que ejecutar, dentro de ese directorio, el comando:

```
lzma -d *.lzma
```

2.2.2. Modificaciones del código

En el código fuente original, proporcionado en el repositorio GitHub del libro base, se han detectado algunos **errores que es necesario corregir**, además de adaptarlo para Python3, para que el código funcione correctamente. En esta actividad se utilizará el código de ejemplo del subdirectorio `IDPanel`, que se obtiene tal como se ha indicado en el apartado 2.2.1. Se deben hacer las siguientes modificaciones:

[3] Alternativamente se puede clonar localmente el repositorio de GitHub con el comando:
`git clone https://github.com/cylance/IntroductionToMachineLearningForSecurityPros.git`

- Fichero: `IDPanel/train_model.py`
  - Editar la línea 4 para sustituir "`cross_validation`" por "`model_selection`", que es el nuevo nombre del módulo en versiones recientes de *scikit-learn*.
  - Editar la línea 33 para sustituir "`y`" por "`*y, **z`", con lo que la línea debe quedar así:
    `warnings.warn = lambda x, *y, **z: x`
  - Editar la línea 45 para sustituir "`training`" por "`testing`", para que el mensaje se corresponda con el conjunto.
  - Editar la línea 109 para sustituir `"w"` por `"wb",` con lo que la línea debe quedar así:
    `with open("bot_model.mdl", "wb") as f:`
- Fichero: `IDPanel/train_lr_model.py`
  - Editar la línea 4 para sustituir "`cross_validation`" por "`model_selection`", igual que en el anterior.
  - Editar la línea 44 para sustituir "`y`" por "`*y, **z`", con lo que la línea debe quedar así:
    `warnings.warn = lambda x, *y, **z: x`
  - Editar la línea 55 para sustituir "`training`" por "`testing`", para que el mensaje se corresponda con el conjunto.
  - Editar la línea 108 para sustituir `"w"` por `"wb",` con lo que la línea debe quedar así:
    `with open("bot_model.lrmdl", "wb") as f:`
- Conversión a Python3: para que el código sea ejecutable con Python3 es necesario modificarlo ejecutando, desde dentro del directorio `IDPanel/`, el siguiente comando[4]:
  `2to3 -w -x dict *.py idpanel`
  De esta forma se modificarán automáticamente los ficheros (mayormente cambios de: `print()`, `range()` y alguna envoltura `list()` de un iterador) y se crearán copias en ficheros `.bak` de los originales, para los ficheros directamente en el directorio y también recursivamente en el subdirectorio `idpanel`.

3. Resultados requeridos

En esta actividad se van a replicar los cálculos de ejemplo del libro base con los datos del repositorio GitHub, para analizarlos y compararlos con los resultados mostrados en el libro. También se propone realizar un análisis similar con una modificación de los datos que elimine una característica importante de clasificación para ver en qué varía la clasificación.

3.1. Cálculos con los datos de ejemplo

En primer lugar, después de realizar la instalación y descarga de los ficheros tal como se ha indicado en el apartado 2, se deben ir ejecutando los mismos comandos (cambiando "`python`" por "`python3`", evidentemente) indicados en el apartado "*Classification Applied to Real-World Security Threats*" del libro base (final del capítulo 2). Es necesario haber realizado **previamente la descompresión de los ficheros .lzma** indicada en el apartado 2.2.1, porque, en caso contrario, el primer *script* (en el apartado "*Data Collection*"), al no encontrar el fichero `prevectors.json`, intentaría volver a descargar de nuevo los datos (todos los *offsets* de todas las *urls* de base), pero todas las direcciones de *BotNets* usadas ya no están activas. Los *scripts* correspondientes se pueden encontrar en el directorio indicado en el apartado 2.2.1.

Como respuesta de este apartado (en el documento de la actividad explicado en el apartado 1.4), se deben indicar los resultados obtenidos en cada caso (*Árboles de decisión* y *Regresión logística*) y compararlos con los correspondientes del libro.

Es importante recordar que los resultados mostrados en el libro se corresponden con un subconjunto de los datos incluidos en el repositorio GitHub, por lo cual los resultados serán algo diferentes numéricamente.

[4]En el comando se usa la opción "`-x dict`", que excluye las correcciones para envolver `dict.keys()` con `list()`, ya que en los casos usados en estos ficheros no es necesario y perjudica la eficiencia.

También se recomienda hacer las pruebas de entrenamiento de los modelos más de una vez (en este caso un par de veces puede ser suficiente), ya que hay cierta aleatoriedad en la inicialización de los algoritmos que puede producir resultados ligeramente distintos.

El análisis y comparación que se pide se debe realizar de forma cualitativa para determinar si son coherentes y compatibles. Es recomendable analizar sobre todo los gráficos producidos y su significado. Se deben incluir explicaciones posibles de las diferencias observadas[5].

3.2. Cálculos eliminando una característica principal

En esta segunda parte se pide realizar un análisis similar, aplicado al mismo conjunto de datos, pero eliminando la principal característica de clasificación obtenida mediante árbol de decisión del anterior apartado. Se trata de observar los resultados de clasificación sin esa característica y compararlos con los obtenidos en el apartado anterior, tanto para árboles de decisión, como para regresión logística.

Para realizar la exclusión de una característica[6], se proponen unos cambios en los ficheros de código siguientes:

- Fichero: `IDPanel/extract_features_from_prevectors.py`
  - Editar la línea 3 para que quede de la siguiente forma:
    `from idpanel.blacklist import *`
  - Insertar las dos líneas siguientes, después de la línea 15 (e indentado al mismo nivel de la 14):
    ```
    if any(bl == line['offset'] for bl in feature_blacklist):
        continue
    ```
- Fichero: `IDPanel/vectorize_with_raw_features.py`
  - Editar la línea 4 para que quede de la siguiente forma:
    `from idpanel.blacklist import *`
  - Insertar las dos líneas siguientes, después de la línea 44 (e indentado dentro de ese bucle "`for`"):
    ```
    if any(bl == dp['offset'] for bl in feature_blacklist):
        continue
    ```
- Fichero: `IDPanel/idpanel/blacklist.py`
  - Añadir a la lista "`feature_blacklist`" (línea 12), la cadena de texto del *offset* correspondiente al primer nodo del árbol de decisión obtenido en la primera parte. De esta forma se ignorarán los vectores con ese rasgo.

Una vez que se vuelvan a extraer las características de los prevectores (ignorando el *offset* indicado), se podrán vectorizar de nuevo los datos. Ambas operaciones se harán de forma análoga a las realizadas en la primera fase.

Después se pide volver a aplicar los algoritmos de entrenamiento o ajuste de la primera parte (árboles de decisión y regresión logística). Al igual que en el apartado anterior, se recomienda probar el entrenamiento más de una vez y ver si hay variaciones.

Como respuesta de este apartado (en el documento de la actividad explicado en el apartado 1.4), se deben indicar:

- El valor de *offset* ignorado y otros cambios realizados en los *scripts*.
- Los resultados de estas pruebas (tanto numéricos como gráficas o diagramas) y los parámetros utilizados para ello.
- Una comparación con los resultados del anterior apartado para analizar qué ha cambiado.
- Se deben explicar en qué se detectan las mejoras o empeoramiento de los resultados, etc., además de si los modelos resultantes siguen siendo capaces de discriminar entre ambas clases.

[5] Recuerde que los ejemplos de *BotNets* del libro, y en los datos del repositorio, son de hace varios años, con lo que las *BotNets* conocidas ya habrán sido eliminadas.

[6] En realidad, los cambios propuestos aquí permiten eliminar varias características u *offsets*. Se ignorarán todos los *offsets* que aparezcan en la variable "`feature_blacklist`".