Actividad 1

Datos del estudiante

- **Nombre:** Sergio
- **Apellidos:** Roselló Morell
- **DNI:** 53632974X
- **email:** Sergio-resello@hotmail.con

Información sobre el entorno

- **Sistema Operativo:** Arch Linux
- **Entorno de escritorio:** dwm
- **Versión de Python:** Python 3.8.2
- **Editor de texto:** NeoVim
- **Generación del documento:** Escrito en MD, compilado a LaTeX con

```
nnoremap <leader>e :! pandoc % -f markdown -t latex -s
-o %:r.pdf<cr>
```

Breve resumen de la actividad a realizar

Respuestas de cada uno de los apartados pedidos en la actividad

Apartado 1 - Cálculos con los datos de ejemplo

En el algoritmo de K-means, únicamente podemos cambiar el número de clusters que queremos generar. En este ejemplo, tiene sentido que lo asignemos a dos, ya que un cluster debería pertenecer a las IP con tráfico estándar y el otro debería pertenecer a las IP con tráfico malicioso.

A partir de los resultados, tenemos que determinar si tienen sentido o no. Este es el verdadero trabajo del analista.

Valores de parámetros con K-Menas En la preparación de los datos de entrada, el analista decide usar únicamente las direcciones IP que tengan mas de 5 entradas en el registro. Esto es útil para buscar un tipo específico de ataque, pero dependiendo del ataque que se quiera buscar, se deberá decidir el número de entradas y valores a revisar. Si el analista busca encontrar un ataque con persistencia en el sistema, no debería contar el número de peticiones, sino el valor de la petición.

En el ejemplo del libro, se establece el hyperparámetro 2, de acuerdo con la lógica descrita anteriormente.

En siguientes iteraciones, he creado un script sencillo que itere desde 2 a 22 clusters para posteriormente analizar los resultados y determinar una solución que englobe los máximos casos de peticiones maliciosas posibles.

Modificaciones de código con K-Menas No ha habido modificaciones de código en la solución guiada.

Se ha generado un script sencillo para realizar la iteración de los clusters

```
#!/bin/bash

COUNTS=20
CLUSTERS=2

while [ $COUNTS -gt 0 ]
do
  python cluster_vectors.py -c kmeans -n $CLUSTERS -i secrepo.h5 -o secrepo-$CLUSTERS.h5
  python stats_vectors.py -i secrepo-$CLUSTERS.h5 > secrepo-$CLUSTERS.log
  python label_notes.py -i secrepo-$CLUSTERS.h5 | grep  70.32.104.50 >> secrepo-$CLUSTERS.lo
  COUNTS=$[$COUNTS-1]
  CLUSTERS=$[$CLUSTERS+1]
done
```

Fallos o errores y solución con K-Menas No he tenido ningún fallo en el cálculo del ejercicio. Lo único relevante es que se ha tenido que cambiar el comando para obtener las estadísticas de:

```
python stats_vectors.py archivo.h5
```

a

```
python stats_vectors.py -i archivo.h5
```

Mis resultados siguiendo la guía del libro base de la asignatura son algo similares al mismo.

Ambos clusters se han agrupado de la siguiente forma:

- **Label 0:** "silhouette" 0.833 con 77.98% pertenencia (las conexiones "normales")
- **Label 1:** "silhouette" 0.242 con 22.02% pertenencia (las conexiones "diferentes" que no necesariamente significan que son ataques, pero son más probables)

Buscando la dirección IP que sabemos que es maliciosa, vemos que efectivamente, se cumple nuestra hipótesis. En nuestro caso, pertenece al grupo 1.

Una nueva revisión de los clusters, esta vez añadiendo más clusters, nos indica que, efectivamente con 12 clusters llegamos al cluster mas preciso con el menor nivel de computación necesario.

```
12 clusters:
  Number of items: 69 (0.69%) (avg silhouette: 0.30414125323295593)
```

A partir del cluster 12 vuelve a haber una subida de imprecisión hasta la separación en 17 clusters. De ahí en adelante, el cluster con la IP maliciosa queda estable con 28 elementos. Estos resultados ya parecen sobreoptimizados, de forma que pienso que la separación el 12 clusters es la más óptima dados los datos de entrada.

Dado que los resultados del algoritmo dependen de la posición inicial a la que se asignan los primeros centroides, el procedimiento se ha realizado tres veces para asegurar que es un resultado correcto. En los tres casos, el mejor resultado surge al dividir el dataset en 12 clusters.

Valores de parámetros con DBSCAN En las primeras pruebas se han usado los mismos parámetros que en el ejercicio guiado del libro base de la asignatura.

Al ver que no estaba siendo efectivo el método clustering, se ha cambiado `epsilon` a valores menores. Este cambio hace que se contemplen dos puntos como que están en un "vecindario" distinto si su distancia (En este caso Euclídea) es superior a `epsilon`. Esto quiere decir, que a menor `epsilon`, mayor la subdivisión en clusters de los datos.

Puesto a que `min_samples` controla la capacidad de crear clusters en relación a los puntos que un centroide tiene a su alrededor, no se quiere reducir demasiado el valor ya que esto hará que se creen clusters de valores irrelevantes.

Modificaciones de código con DBSCAN No se ha modificado el código de los scripts proporcionados por el repositorio del libro de la asignatura.

Fallos o errores y solución con DBSCAN Una de las curiosidades que se han dado es que, a diferencia del resultado que le aparece a los autores del libro, el algoritmo DBSCAN me indica que hay dos o tres clusters en el grupo proporcionado. Esto es un resultado un tanto extraño y no consigo averiguar a qué se debe.

```
python cluster_vectors.py -c dbscan -e 0.5 -m 5 -i secrepo.h5 -o secrepo.h5
Label -1 has 18 samples
Label 0 has 9977 samples
Label 1 has 5 samples
python cluster_vectors.py -c dbscan -e 0.6 -m 5 -i secrepo.h5 -o secrepo.h5
Label -1 has 16 samples
Label 0 has 9984 samples
```

Con los resultados obtenidos en el análisis anterior (Provisto por el libro) no se clasifican bien las IP maliciosas. Como ejemplo, la IP 70.32.104.50 aparece en el cluster 0, junto con la mayoría de IP en el sistema.

Tras probar con una variedad de valores, tanto de `min_samples` como de `epsilon`, teniendo en cuenta nuestros datos, pienso que nos interesa encontrar un número

de clusters prudente relacionando puntos cercanos entre si. Esto se traduce a aumentar `min_samples` a 8 y reducir `epsilon` a 0.1

Estos datos no se ven respaldados puesto que la dirección IP 70.32.104.50 se encuentra en el grupo 0, que tiene 22 otras IP's de las cuales muchas tienen comportamientos distintos y no parecen seguir un comportamiento predecible.

Por ejemplo, otra dirección IP con intención maliciosa es la `192.187.126.162`, que pertenece al cluster -1.

Apartado 2 - Calculo con otros datos diferentes

Valores de parámetros

- Los casos y parámetros de clustering que se han probado inicialmente (de K-Means o DBSCAN) y porque.

Modificaciones de código

- Los resultados de estadísticas de las primeras pruebas que se han usado para decidir otros valores de los parámetros a probar y como se han usado para elegirlos.

Fallos o errores y solución

- Si se han podido encontrar posibles clusters con IPs sospechosas de potenciales ataques, y que características se observan en los resultados que se han obtenido en cada caso al analizar los registros correspondientes.

Comentarios y opiniones

Dificultades/Problemas encontradas

Programas/Ayudas utilizadas

Comentarios sobre la realización de la actividad

Bibliografía