Luis Daniel Rivero Sosa
A01374527

Reporte:

1- Procedimiento de limpia de datos

Para este punto se reemplazaron los datos pertenecientes a cadenas de texto por una referencia simbólica en número. El archivo con los datos limpios usados se puede encontrar en el archivo **“examen.csv”** dentro de mi carpeta entregada.

Para este paso se hicieron los siguientes cambios en las columnas:

Columna protocol_type:

- tcp = 1
- icmp = 2
- udp = 3

Columna Service:

- private = 1
- ftp_data = 2
- eco_i = 3
- telnet = 4
- http = 5
- smtp = 6
- ftp = 7
- ldap = 8
- pop_3 = 9
- courier = 10
- discard = 11
- ecr_i = 12
- imap4 = 13
- domain_u = 14
- mtp = 15
- systat = 16
- iso_tsap = 17
- other = 18
- csnet_ns = 19
- finger = 20
- uucp = 21
- whois = 22
- netbios_ns = 23
- link = 24
- Z39_50 = 25
- sunrpc = 26
- auth = 27
- netbios_dgm = 28
- 21_path = 29
- netbios_ns = 30
- vmnet = 31
- domain = 32
- name = 33
- pop_2 = 34
- 5_443 = 35
- supdup = 36
- gopher = 37
- bgp = 38
- time = 39
- remote_job = 40
- kshell = 41
- IRC = 42
- printer = 43
- ssh = 44
- nntp = 45
- day39 = 46
- ctf = 47
- urp_i = 48
- host33s = 49
- echo = 50
- login = 51
- exec = 52
- sql_net = 53
- shell = 54
- pm_dump = 54
- netstat = 55
- netbios_ssn = 56
- tim_i = 57
- nnsp = 58
- rje = 59
- efs = 60
- X11 = 61
- k51 = 62
- ntp_u = 63
- t7_u = 64

Columna flag:

- REJ = 1
- SF = 2
- RSTO = 3
- S0 = 4
- SH = 5
- RSTR = 6
- S1 = 7
- S3 = 8
- s2 = 9
- OTH = 10

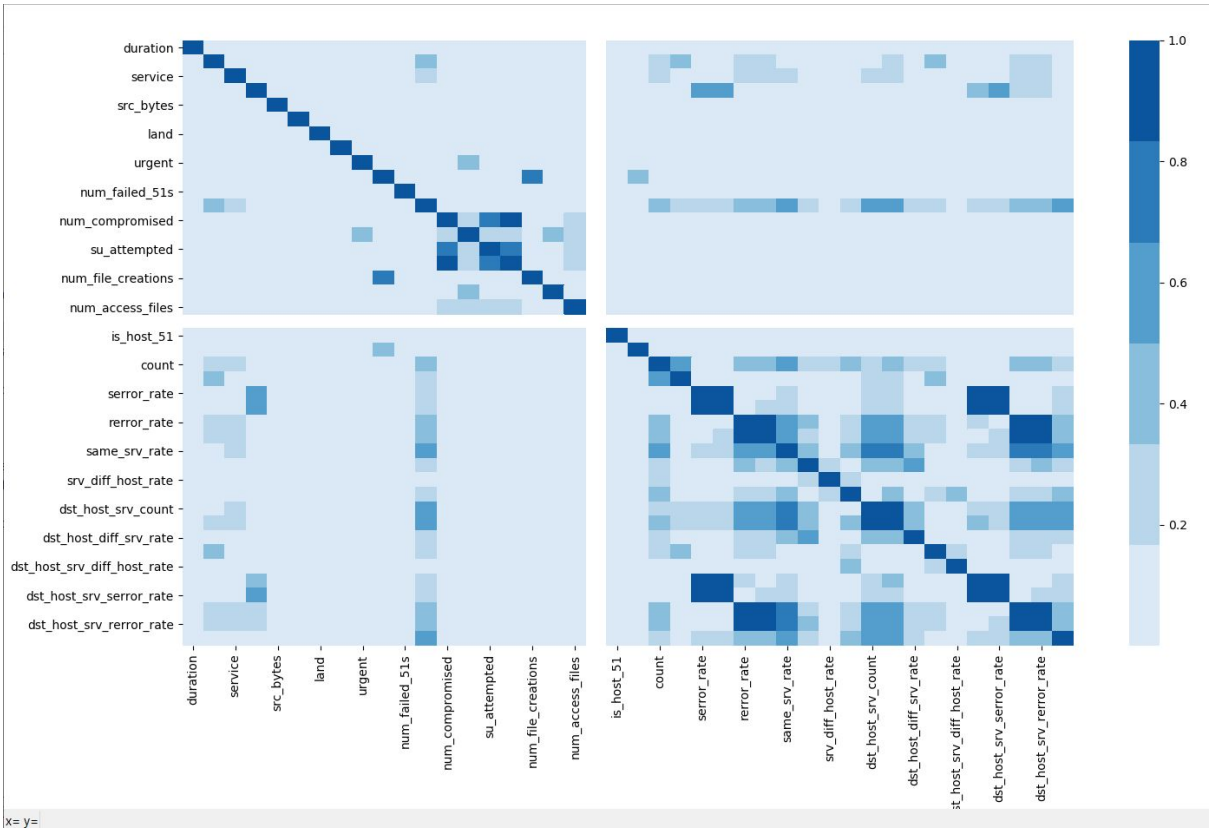
Luis Daniel Rivero Sosa
A01374527

Columna class:

- anomaly = 0
- normal = 1

2- Mapa de calor HeatMap

Para este paso se uso el archivo llamado **“HeatMap.py”**. El resultado del mapa de calor fue el siguiente:



Donde se identificaron las siguientes variables correlacionadas:

- rerror_rate
- srv_rerror_rate
- same_srv_rate
- dst_host_srv_serror_rate
- dst_host_rerror_rate
- dst_host_srv_count
- dst_host_same_srv_rate

Luis Daniel Rivero Sosa
A01374527

3- Diferencias en los valores de F1 usando KNN variando k

Para elegir la mejor K, hice 10 folds, cada uno con 5% de datos de prueba y 95% de datos de entrenamiento con respecto al "dataset" original. Esto usando el código **"crearArchivosAleatorios.py"**. Posteriormente use el código **"10folds.py"** para promediar el F1 con diferentes K´s. Los resultados obtenidos fueron los siguientes:

```
Resultados al usar k = 1


Accuracy:
0.735359361136

f1
0.733171170556
```

```
Resultados al usar k = 3


Accuracy:
0.731144631766

f1
0.728354084283

Resultados al usar k = 5


Accuracy:
0.731588287489

f1
0.728809328627


Resultados al usar k = 15


Accuracy:
0.724489795918

f1
0.720271157757
```

Este procedimiento se realizó con 5% de datos de prueba y no 20% debido a que la capacidad computacional para ejecutar este código se mostró muy tardado con la capacidad de una computadora promedio. Sin embargo con dichas iteraciones se seleccionó K=5 cómo mejor opción.

Posteriormente se ejecutaron 10 folds con 20% de datos de prueba con K = 5 en Knn y el resultado fue el siguiente:

```
Resultados al usar k = 5


Accuracy:
0.974621001385

f1
0.970033505035
```

4- Comparación de f1 del mejor KNN vs MLP

Para obtener el MLP se uso el código **"MLP.py"**. Donde se ejecutó el algoritmo con 20% de datos de prueba elegidos de forma aleatoria y el resultado fue el siguiente:

```
[[2493   44]
 [  86 1886]]
             precision    recall  f1-score   support

          0       0.97      0.98      0.97      2537
          1       0.98      0.96      0.97      1972

avg / total       0.97      0.97      0.97      4509
```

Tomando esto en cuenta **el f1 del algoritmo Knn fue igual al del MLP (0.97)**

La diferencia más notable es que el algoritmo MLP tomo menos del 5% de tiempo en ejecutarse a comparación del Knn

5- Variables eliminadas

Las variables eliminadas fueron las mismas variables correlacionadas obtenidas en el HeatMap:

- rerror_rate
- srv_rerror_rate
- same_srv_rate
- dst_host_srv_serror_rate
- dst_host_rerror_rate
- dst_host_srv_count
- dst_host_same_srv_rate

Luis Daniel Rivero Sosa
A01374527

Este conjunto de datos limpio se puede encontrar como **“examenLimpio.csv”**.

6- F1 obtenida quitando las variables

Para este paso se utilizó el algoritmo de MLP por que el f1 era igual al del Knn, sin embargo el tiempo de ejecución fue mucho menor. El resultado fue el siguiente:

```
[[2505   33]
 [  68 1903]]
             precision    recall  f1-score   support

          0       0.97      0.99      0.98      2538
          1       0.98      0.97      0.97      1971

avg / total       0.98      0.98      0.98      4509
```

En este caso se puede observar que el f1 y la precisión mejoraron 0.01 a comparación de el uso del conjunto de datos original.