UNIVERSIDAD PANAMERICANA
Campus México

UNIVERSIDAD PANAMERICANA

Materia: Gestión de Análisis y Diseño de Comercialización(COM145)

Profesor: Sarahí Aguilar González

Fecha de entrega: 24 mayo, 2022

Ciclo: 1222

Nombre del proyecto: Identificación de Patrones de URLs Maliciosos

Miembros del Equipo		
ID	**Nombre**	**Carrera**
0213358	Flores Peregrina, Ricardo Ariel	LTISC
0212614	Mayen Soto, Esteban	LTISC

Rúbricas				
ID	**2-social**		**7-knowledge**	
	D	**C**	**A**	**JI**

Abstract — La oportuna detección de URLs maliciosas tendría un gran valor ante la sociedad ya que muchos de los usuarios de internet navegan de una manera inconsciente y más tarde se encuentran pagando por ello. Por ello es que a través de nuestro modelo y herramientas de Data Science, buscamos categorizar qué es lo que realmente distingue a un URL inofensivo de uno que podría tener algún fin malicioso.

Palabras clave — enlace, URL, liga, malware

Introducción

Pregunta de investigación

“¿Los links que recibimos en el día a día con patrones anormales pueden ser confiables?”

Día a día, cada persona individualmente tiene miles de interacciones con internet de diferentes maneras, pero la más común de ellas por medio de ligas que pueden recibir o simplemente acceder porque les aparecen en algún lado. Aquí es donde entra la problemática que queremos resolver con nuestro proyecto, “¿como sabemos en que si podemos confiar y en que no?” Para ello es que hemos decidido por medio del uso de herramientas de Data Science, identificar patrones en ligas de acceso a direcciones de Internet (URLs) para así poder identificar características particulares en ligas que podrían llevar a sitios con sitios maliciosos o directamente a la descarga de un malware hacia los dispositivos personales de los navegantes de internet.

Desarrollo

Haciendo uso de las herramientas y aplicaciones de Data Science es que planeamos llegar a un modelo de clasificación de patrones para poder tener una mira más angosta sobre que distingue a un enlace posiblemente malicioso de un enlace benigno o inofensivo, que cumple su propósito de dirigir una página o contenido que los usuarios buscan. Dentro de nuestra investigación nos topamos con los distintos tipos de malware que se pueden alojar dentro de una URL. La cantidad de usuarios que tienen una conciencia sobre qué peligros pueden existir en Internet es realmente poca, comparada con la cantidad de gente que navega sin el menor de los cuidados, que acceden a cualquier tipo de enlace y que más adelante se encuentran bajo la situación de que han caído ante un tipo de malware que busca hacer mal uso con su información o que simplemente busca dañar sus equipos personales.

Por medio de una base de datos de URLs clasificados con sus categorías respecto a malware, es que haremos uso de técnicas y herramientas de Data Science para realizar análisis y poder llegar a una conclusión donde podamos identificar patrones particulares que se podrían encontrar en un URL malicioso. Como bien sabemos, no todo lo que se encuentra en internet es bueno siempre, y es por ello que uno de los métodos más sencillos para la propagación de malware es por medio de links, ya que basta con que el usuario de click en el enlace que reciba y es más que suficiente para que un malware tome posesión de su dispositivo y con él, la información personal y sensible que pueda contener.

Comenzamos nuestro análisis tomando como base, un dataset que contiene aproximadamente un millón de links maliciosos con sus categorías dentro del malware, así como también contiene benignos para ayudarnos a tener una comparativa clara de como distinguir por medio de patrones si un link contiene malware o es un sitio inofensivo.

Para tener un manejo mucho mas eficaz y no tener que estar contando con tener el dataset de manera local, optamos por montar dicho dataset en un Kaggle, con el fin de simplemente llamarlo por medio de donde se encuentra alojado. Una vez seleccionado nuestro dataset para trabajar, previamente tuvimos que realizar una limpieza de valores **nulos** y algunos valores extra que decidimos excluir del análisis.

```
data = pd.read_csv('/content/Dataset_BM_1.csv')
data.head()
```

	URL	TIPO
0	http://66.208.203.190:36841/malware.a	malware
1	http://58.255.129.35:53862/malware.a	malware
2	http://60.25.156.155:47183/malware.m	malware
3	http://192.72.17.236:35284/malware.a	malware
4	http://27.41.38.130:50541/malware.m	malware

```
data.info()
```

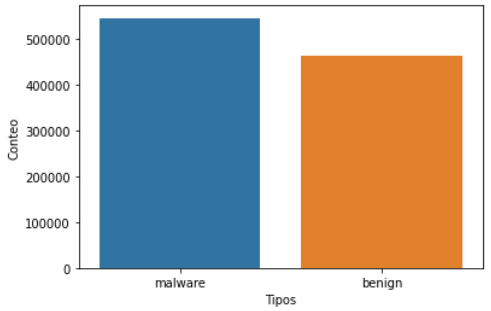
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1008870 entries, 0 to 1008869
Data columns (total 2 columns):
 #   Column  Non-Null Count    Dtype 
---  ------  --------------    ----- 
 0   URL     1008870 non-null  object
 1   TIPO    1008870 non-null  object
dtypes: object(2)
memory usage: 15.4+ MB
```

Al tener nuestro dataset listo para trabajarlo, lo procesamos en un notebook de Jupyter y por medio de librerías para el manejo de los datos fue como pudimos tener visualizaciones más claras de los datos con los que vamos a trabajar. Por medio de las visualizaciones tuvimos una noción mucho más clara sobre los datos que se contienen en el dataset y con los que trabajamos para entrenar a nuestro modelo. Nuestro dataset es bastante sencillo, ya que aunque tiene muchos datos, solo cuenta con 2 columnas por registro, siendo uno el **URL** y el otro **TIPO,** que es el tipo de URL que es ese registro.

```
count = data.TIPO.value_counts()
count
```

```
malware    545389
benign     463481
Name: TIPO, dtype: int64
```

```
sns.barplot(x=count.index, y=count)
plt.xlabel('Tipos')
plt.ylabel('Conteo');
```



Como se mencionó anteriormente, el dataset contiene tanto URLs con malware, así como unos ejemplares con fines benignos. El tener estos dos tipos de URLs en nuestro proyecto nos da una visión más clara de que podrá nuestro modelo identificar patrones de un link malicioso de uno inofensivo. Para el tratamiento de datos hicimos investigaciones y ya existían ciertas librerías las cuales nos ayudan a realizar un mejor análisis de los datos algunas de ellas fueron urlparse, get_tld, is_tld

```
rem = {"Categoria": {"benign": 0, "malware": 1}}
data['Categoria'] = data['TIPO']
data = data.replace(rem)
```

```
data['URL_LEN'] = data['URL'].apply(lambda x: len(str(x)))
```

```
def process_tld(URL):
    try:
        res = get_tld(URL, as_object = True, fail_silently=False,fix_protocol=True)
        pri_domain= res.parsed_url.netloc
    except :
        pri_domain= None
    return pri_domain
```

```
data['DOMAIN'] = data['URL'].apply(lambda i: process_tld(i))
```

```
data.head(200)
```

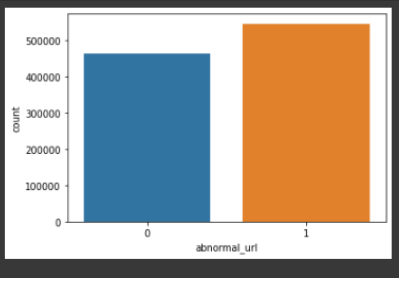
	URL	TIPO	Categoria	URL_LEN	DOMAIN
0	http://66.208.203.190:36841/malware.a	malware	1	37	None
1	http://58.255.129.35:53862/malware.a	malware	1	36	None
2	http://60.25.156.155:47183/malware.m	malware	1	36	None
3	http://192.72.17.236:35284/malware.a	malware	1	36	None
4	http://27.41.38.130:50541/malware.m	malware	1	35	None
...	...	...	...	...	...
195	http://164.163.25.165:41491/bin.sh	malware	1	34	None
196	http://37.120.222.60/mysite/catimages/243.malware	malware	1	49	None
197	http://37.120.222.60/mysite/catimages/244.malware	malware	1	49	None
198	http://37.120.222.60/mysite/catimages/242.malware	malware	1	49	None
199	http://37.120.222.60/mysite/catimages/246.malware	malware	1	49	None

200 rows × 5 columns

```
def abnormal_url(URL):
    hostname = urlparse(URL).hostname
    hostname = str(hostname)
    match = re.search(hostname, URL)
    if match:
        #hay una relacion
        return 1
    else:
        #no se encontre relacion
        return 0
```

```
[24] data['abnormal_url'] = data['URL'].apply(lambda i: abnormal_url(i))
```

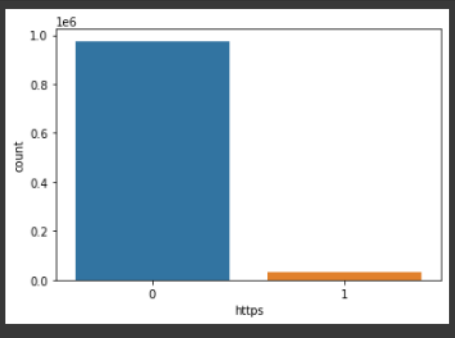
```
[25] sns.countplot(x='abnormal_url', data=data);
```




```
[26] def httpSecure(URL):
        htp = urlparse(URL).scheme
        match = str(htp)
        if match=='https':
            #hay una relacion
            return 1
        else:
            # no hay ninguna relacion
            return 0
```

```
[27] data['https'] = data['URL'].apply(lambda i: httpSecure(i))
```

```
[28] sns.countplot(x='https', data=data);
```




```
def digit_count(URL):
    digits = 0
    for i in URL:
        if i.isnumeric():
            digits = digits + 1
    return digits
```

```
[30] data['digits'] = data['URL'].apply(lambda i: digit_count(i))
```

```
[31] def letter_count(URL):
        letters = 0
        for i in URL:
            if i.isalpha():
                letters = letters + 1
        return letters
```

```
[32] data['letters']= data['URL'].apply(lambda i: letter_count(i))
```

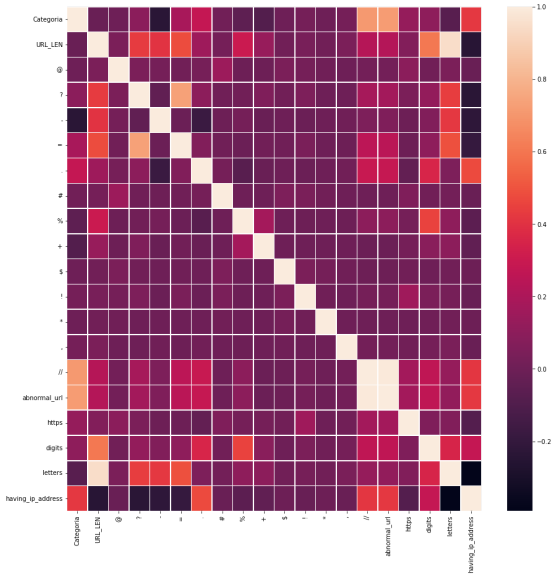
```
[33] def having_ip_address(URL):
        match = re.search(
            '(([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.(
            '([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\/)|'  #identificador de IPV4
            '(([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.(
            '([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\/)|'  # identificador de IPv4 con puer
            '((0x[0-9a-fA-F]{1,2})\\.(0x[0-9a-fA-F]{1,2})\\.(0x[0-9a-fA-F]{1,2})\\.(0
            '(?:[a-fA-F0-9]{1,4}:){7}[a-fA-F0-9]{1,4}|'
            '([0-9]+(?:\.[0-9]+){3}:[0-9]+)|'
            '((?:(?:\d|[01]?\d\d|2[0-4]\d|25[0-5])\.){3}(?:25[0-5]|2[0-4]\d|[01]?\d\d
        if match:
            return 1
        else:
            return 0
```

```
[34] data['having_ip_address'] = data['URL'].apply(lambda i: having_ip_address(i))
```

```
[35] data['having_ip_address'].value_counts()

0    832348
1    176522
Name: having_ip_address, dtype: int64
```

Una vez se conoce el tipo de datos que se contienen en el dataset, es cuando podemos comenzar a identificar patrones que podrían componer un URL malicioso. Para identificar componentes de un URL malicioso, utilizamos identificación binaria para así ir poco a poco dentro del dataset resaltando patrones característicos de estos links maliciosos. Así mismo con el tratamiento de los datos, se fueron agregando columnas que nos podrían dar más detalles respecto a cada registro dentro del dataset, tales como longitud, si cuentan con un dominio, el número distintivo para categorizar de manera binaria si el link es benigno o maligno, etc. Así como un filtro para detectar que un link tuviera una estructura anormal y que pudiera ser evidente para nuestro análisis. También otro factor que buscamos resaltar dentro de los URLs era el hecho de si este provenía de un dominio con el tag **https,** que es un buen factor a resaltar ya que usualmente ese tag solo se le da a sitios que tienen certificados de seguridad y de igual manera este campo fue agregado a nuestro dataset para así utilizarlo en el entrenamiento de nuestro modelo. Y por último uno de los últimos campos que agregamos al dataset fue el factor de cuántos dígitos, letras y si contenían directamente en el URL alguna dirección IP.

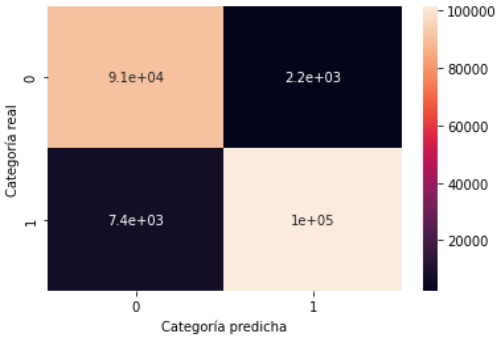


Una vez teniendo los identificadores dentro de nuestro dataset, es que podemos armar nuestra matriz de correlación y así poder ver más a detalle cuáles serán los valores que nos ayudarán dentro de nuestra clasificación.

Para la implementación del modelo decidimos elegir "Árboles de decisión" ya que estamos manejando el proyecto como un modelo supervisado en el cual nosotros le decimos cuales son los URL´s que son benignos o malignos.

En nuestro caso antes de trabajar con el modelo debemos de definir la variable dependiente (categoría) y las independientes (URL,tipo,dominio) después de esto vamos a dividir el conjuntos de datos para el entrenamiento posteriormente debemos de crear el objeto de SVM (Support Vector Machine) que va a ser el de Decision Tree Classifier, una vez con esto vamos a ajustar el modelo con el entrenamiento de X y Y, una vez con esto haremos la predicción de los valores de la variable independiente, una vez con esto vamos a realizar una matriz de confusión con y_test y y_predict, una vez con esto haremos la gráfica de la matriz de confusión donde podremos ver la clasificación en 0 o 1, si el url que estamos trabajando es benigno o maligno, con esto obtenemos lo siguiente:

```
array([[ 90554,   2230],
       [  7416, 101574]])
```



Aquí podemos ver las cuatros opciones que surgen:

1. El URL es benigno y el modelo lo clasificó como benigno (+) . Esto sería un verdadero positivo o VP .
2. El URL es maligno y el modelo lo clasificó como maligno (-) . Este sería un verdadero negativo o sea un VN.
3. El URL es maligno y el modelo lo clasificó como benigno (-) . Éste sería un error tipo II o un falso negativo o FN.
4. El URL es benigno y el modelo lo clasificó como maligno (+) . Este es un error tipo I, o un falso positivo o FP.

Para finalizar la clasificación debemos de calcular la metricas de calidad que son Accuracy, Precision, Recall y F1-Score y obtenemos los siguientes:

```
Accuracy: 0.9521940388751772
Precisión: 0.9785172055026782
Recall: 0.9319570602807598
F1-score: 0.9546697745237178
```

- El accuracy es la relación entre las observaciones correctamente clasificadas y todas las observaciones clasificadas.
- La precisión es la relación entre las observaciones correctamente clasificadas positivas y todas las observaciones clasificadas positivas.
- El recall es la relación entre las observaciones correctamente clasificadas positivas y todas las observaciones realmente positivas.
- El F1-score es la media armónica de la precisión y el recall.

Conclusiones

Con el profundo análisis de nuestro proyecto, podemos concluir que nuestro modelo funcionó de manera exitosa para determinar y diferenciar URLs maliciosas de aquellas inofensivas por medio de las características que pueden componer a un URL con malware. Las herramientas y técnicas utilizadas para Data Science son realmente útiles pero requieren de tener un poco de conocimiento sobre ellas para poder ser aplicadas de manera propia y con el fin que les queremos dar. Más adelante y por medio de las características definidas, se podría buscar implementar nuestro modelo en sistemas generales como aquellos de mensajería (email o plataformas de comunicación) para que cualquier URL que se reciba en el dispositivo de la persona, sea comparado brevemente y en caso de ser una sospecha de malware, avisarle al usuario y que bajo su propio riesgo acceda al link. También otro factor que no descartamos para una futura mejora, sería comparar nuestro modelo contra algún otro que pudiese existir respecto a este tema.

Bibliografía

- **IBM. (2021, 12 marzo).** ***Supervised vs. Unsupervised Learning: What's the Difference?*** **https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning**

- ***Malware & ransomware*****. (2018, 5 enero). Australian Competition and Consumer Commission. https://www.scamwatch.gov.au/types-of-scams/threats-extortion/malware-ransomware#:%7E:text=Malware%20scams%20work%20by%20installing,details%20and%20commit%20fraudulent%20activities**

- **Security, S. (2021, 22 abril).** ***What Is a Malicious URL? (And How You Can Avoid Them)*****. Savvy Security. https://cheapsslsecurity.com/blog/what-is-a-malicious-url/**