Universidad del Valle de Guatemala

Facultad de Ingeniería

Security Data Science

Sección 10

Proyecto 2

Detección de infección por Malware

Presentado por:

Cristina María Bautista Silva (161260)

Andrea Abril Palencia Gutierrez (18198)

Guatemala, 23 de abril 2022

Abstract

The paper discusses the importance of acknowledging if a Windows device is infected, after discovering if indeed it's infected, to take preventive measures. The models that are used to predict if a device is infected are Logistic Regression and Decision Tree Classifier, both are trained with a sample of the given train.csv document and predicted with a sample of the given test.csv document. The performance of both models are evaluated, and are represented with the area below the curve of the ROC curve.

Introducción

En los últimos años se ha visto el aumento de usuarios de computadoras, siendo la mayoría de sistemas operativo Windows, pero el uso de estos dispositivos ha tenido algunos pormenores, siendo un problema la infección de los ordenadores. Esto se puede dar por una gran cantidad de escenarios, algunos ejemplos serían, dar click en publicidad sospechosa mientras se navega en internet. Otro podría ser, el compartir USB infectados y luego conectarlos a máquinas no infectadas.

El mayor problema se encuentra en que consecuencias puede suceder al tener un dispositivo infectado, robo de información y luego exigir una recompensa para recuperar esa información, distribución de información personal en el internet o incluso borrar información importante del ordenador, son algunos ejemplos.

Por ello es importante conocer cómo utilizar los dispositivos de una manera correcta, al igual que entender cuando un dispositivo pueda estar infectado por malware, además de cuáles dispositivos son más vulnerables a qué familias de malware. Y si se detecta que un dispositivo se encuentra infectado qué medidas preventivas realizar para evitar que dañen la integridad del usuario propietario del dispositivo.

Marco teórico

La técnica de regresión logística es uno de los métodos que se utiliza comúnmente en escenarios donde se desea predecir si existe la presencia o falta de una característica que se está evaluando. La diferencia entre este modelo y el de regresión lineal, se da en que la variable dependiente es dicotómica, lo que significa que solo pueden ser entre dos valores, normalmente entre 0 o 1, demostrando la falta o la presencia, respectivamente. También se le puede llamar a la variable de tipo cualitativa o categórica. Esto en función de una variable cuantitativa. Se utiliza más en escenarios de investigaciones de un análisis discriminante.

La técnica de clasificación por árboles de decisiones, tiene la misma característica de obtener la probabilidad de obtener un resultado determinado, el proceso que maneja este método es la agrupación de observaciones con características similares a la variable dependiente. Pero este mismo modelo aplica una serie de reglas, se van tomando esas diferentes reglas en pequeñas divisiones, que luego si tienen más características se van subdividiendo, así es como se van proporcionando los grupos.

El desbalance de datos es otro problema que se debe abordar a la hora del preprocesamiento de la data, ya que si se entrena a un modelo que tiene clases con muy pocas muestras, el resultado ya no será el ideal, no logrará tener buenos resultados con esas clases minoritarias. Por lo que existe la librería de imblearn, que tienen varias alternativas de cómo lograr hacer un balanceo de datos. Se debe tomar en cuenta que se pueden hacer dos tipos de balanceo. El primero siendo oversampling, lo que quiere decir que la clase mayoritaria tiene un número x más grande que el del resto de clases, por lo que al aplicar el oversampling hará que las otras clases suban al número x y todas las clases en efecto estén balanceadas. Por otro lado, existe el undersampling, que la clase que tenga menos muestras con un número y, se le aplicará undersampling a todas las clases y se volverán con menos muestras todas, para que tengan la misma cantidad.

Propuesta de metodología de los modelos (explicación de fases)

Previo al preprocesamiento del trabajo, se leyó el documento del proyecto que tenía un listado de las columnas, con una breve descripción, que conforman tanto el documento csv de train como en el de test, por lo que se discutió qué columnas mantener para el entrenamiento del proyecto.

El siguiente paso fue cargar el documento de train, se crearon en diferentes notebooks el procesamiento de documento de train y de test. Lo primero fue hacer un drop de las columnas que no se encontraron necesarias, se revisaron si entre las columnas seleccionadas existían valores nan, y se encontraron que si. Pero incluso con una cantidad menor de columnas el mismo dataset, seguía demasiado grande para procesarlo con los recursos de máquina disponible, por lo que en este caso, se tomó la decisión de tomar un muestreo del dataset. Se decidió tomar una muestra de un millón por cada dataset (de train y de test), se guardaron en un documento csv llamado sample_train y sample_test y se creón un nuevo notebook para pre procesar la información de cada nuevo csv.

Para los nuevos notebooks, se chequearon en que columnas existían valores nan, ya que se debía aplicar alguna técnica, para cambiar esos valores. Se procesó cada columna de la siguiente manera. El primer paso era usar la

función de value_counts() para obtener los valores que más repetían, esto nos llevó a cambiar casi todas las columnas, ya que en algunas existían muchos valores que solo se repetían una vez, por lo que se toma los top 5 valores más repetidos y se añadían a una lista. Lo siguiente era crear un ciclo for que iterara por toda la columna, y si miraba una condición, si el valor no se encontraba en la lista previamente creada, se tenía que cambiar de valor. Este nuevo valor se determinaba con un random.choice() que tenía la lista de los top 5, por lo que, en cada vuelta del ciclo se volvía a escoger aleatoriamente un nuevo valor. Con el nuevo valor creado, se sustituía el viejo valor, y así sucesivamente. Y este mismo procedimiento con todas las tablas creadas. Lo siguiente, era convertir las columnas con tipo object a tipo int64. Se realizó con la función replace() y con diccionarios, ya que es mucho más fácil trabajar con números que con letras, a la hora de obtener métricas u otras funciones con un objetivo más numérico como obtener promedios, entre otros. Para guardar los datasets con todos los cambios se crearon como segundos datasets. y se creó un nuevo notebook.

Se importaron los ya actualizados datasets al nuevo notebook, pero faltó balancear el archivo de train, ya que la columna target no está balanceada, por poco, pero si se debe balancear. Al balancear se encuentra el problema de que el shape es mayor a los otros, por lo que se debe sacar una muestra que tenga el mismo tamaño para ser utilizada.

Lo siguiente es aplicar los modelos de Logistic Regression y Decision Tree Classifier, hacer fit con “x” y “y” del dataset de train, y para las predicciones con el “x” del dataset de test. El siguiente paso, es obtener el accuracy score, la matriz de confusión, el reporte de clasificación y obtener la gráfica de roc_curve(). El mismo procedimiento se aplicó en ambos modelos. Y se concluye el procedimiento.

Resultados del muestra de Dataset

```
HasDetections
1                500693
0                499307
dtype: int64
```

Resultados del muestra de Dataset balanceado

```
HasDetections
0                500000
1                500000
dtype: int64
```

Resultados del modelo de Logistic Regression

Accuracy del modelo

```
Mean Accuracy: 0.511 (0.000)
```

Matriz de confusión

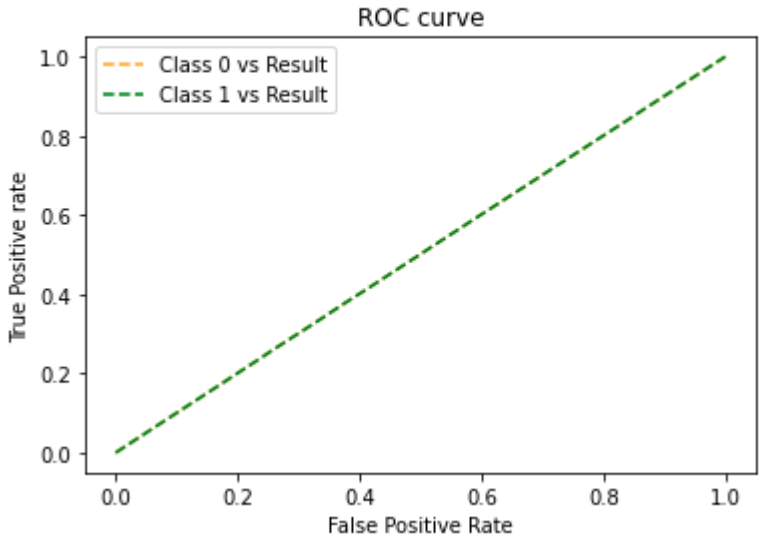
```
array([[255727, 244273],
       [255416, 244584]])
```

Reporte de clasificación

	precision	recall	f1-score	support
0	0.50	0.51	0.51	500000
1	0.50	0.49	0.49	500000
accuracy			0.50	1000000
macro avg	0.50	0.50	0.50	1000000
weighted avg	0.50	0.50	0.50	1000000

Gráfica de la curva de ROC

```
<matplotlib.legend.Legend at 0x133a6d460>
```



Resultados del modelo de Decision Tree Classifier

Accuracy del modelo

```
Mean Accuracy: 0.531 (0.000)
```

Matriz de confusión

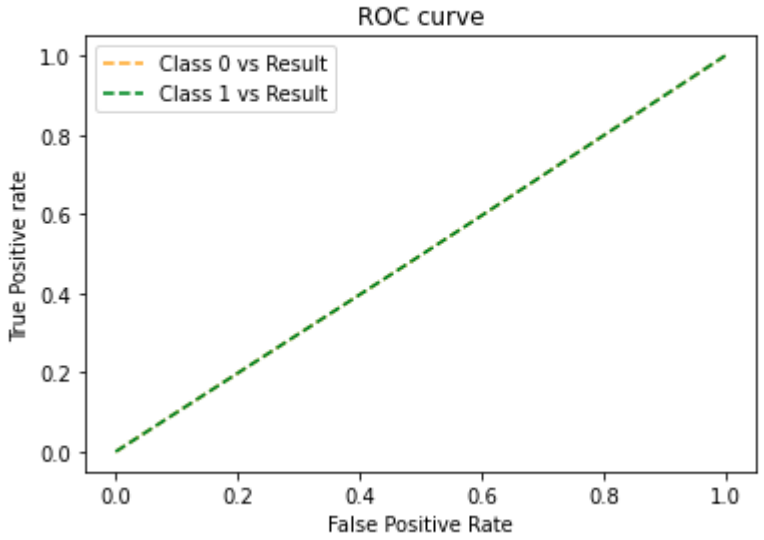
```
array([[264178, 235822],
       [266411, 233589]])
```

Reporte de clasificación

	precision	recall	f1-score	support
0	0.50	0.53	0.51	500000
1	0.50	0.47	0.48	500000
accuracy			0.50	1000000
macro avg	0.50	0.50	0.50	1000000
weighted avg	0.50	0.50	0.50	1000000

Gráfica de la curva de ROC

```
<matplotlib.legend.Legend at 0x13423f490>
```



Conclusión

Con los resultados obtenidos, se puede concluir que, se necesita tomar una muestra más grande del dataset para trabajar, esto se debe a que ambos modelos tienen un accuracy bajo muy similar, si existiera una diferencia mayor entre modelos, puede ser que las features que se utilizaron no eran las ideales.

En el preprocesamiento, se cambiaron varias columnas que tenías demasiados valores únicos, un cambio que puede cambiar el accuracy para ambos modelos, puede ser utilizar distintas técnicas para probar si esto pudiese mejorar el performance de ambos modelos.

Eliminar algunas columnas o agregar otras columnas que tengan información valiosa, pre procesar la información, revisión de valores nan, para darle mayor valor el dataset y el entrenamiento sea mejor y más general para las clases.

Recomendaciones

Si se desea replicar el proyecto, se recomienda tener una computadora con más recursos, para trabajar con todo el dataset o trabajar con muestras más grandes de ambos datasets.

Si se tienen los recursos computacionales, se puede tomar una muestra mayor, y si hay desbalance de datos, aplicar una técnica de undersampling, ya que no se replican datos para balancear, al contrario, trabajar con los que se tiene en un escala un poco menor, pero mayor que en este trabajo.

Referencias

Davilac. (2020). *Qué son los árboles de decisión y para qué sirven*. Máxima Formación. https://www.maximaformacion.es/blog-dat/que-son-los-arboles-de-decision-y-para-que-sirven/#id1

IBM. (2021). *Regresión Logística*. https://www.ibm.com/docs/es/spss-statistics/SaaS?topic=regression-logistic

Rodrigo, J. A. (2016). *Regresión logística simple y múltiple*. Ciencia de Datos. https://www.cienciadedatos.net/documentos/27_regresion_logistica_simple_y_multiple

Unir. (2021, 19 octubre). *Árboles de decisión: en qué consisten y aplicación en Big Data*. https://www.unir.net/ingenieria/revista/arboles-de-decision/