# Proyecto 2 Prediccion de infeccion por malware

César Rodas 16776
Mario Perdomo - 18029
Universidad del Valle de Guatemala
Ciencias de la Computación y T.I
Security Data Science

***Resumen*—The following work describes the process, models and results of the training of two multi-class classification models (Random forest and Naive Bayes) for the predictions of infection regarding malware. The main objective is to predict the probability of a Windows OS machine to be infected by various malware families, based on the machine's properties. The data is provided by Microsoft and it will be used to train our models to predict the outcomes. The outcome of this project was that the Random Forest had the highest accuracy, which was 60 %, in terms to detect malware. However, during our memory usage efficiency, it was discovered that utilizing the method PCA (Principal Component Analysis) would improve both parametrics in regards to the highly-volume dataset.**

***Index Terms*—Naive Bayes - Random Forest - Attacks - Malware - Infection - EDA - Roc's Curve - PCA - KFolds.**

## I. Introducción

El siguiente trabajo presenta las distintas fases en el proceso de procesamiento, selección, entrenamiento y analisis de metricas de dos distintos modelos de calsificación sobre un data set proporcionado por el proyecto proporcionado de Microsoft. Es un evento organizado por la empresa Microsoft dueña del sistema operativo Windows, en el cual generar modelos de Machine Learning (ML) con el objetivo principal de predecir si una maquina será infectada por Malware basado en las características del equipo con el fin de poder tomar medidas preventivas antes que suceda. Para el evento Microsoft provee la data de la telemetría con las propiedades de la computadora, información de máquinas infectadas y reportes de amenazas generados con Windows Defender que es el software de defensa que se instala en equipos con SO Windows.

Para este proyecto se generarán dos modelos de ML, Random Forest y Naive Bayes, y para ello se utilizan dos datasets proporcionados por Microsoft para el evento, el primero es un dataset de observaciones con 8,921,483 de registros que servirá para generar los modelos y el segundo un dataset con 7,853,253 registros como datos de prueba para comprobar las predicciones de los modelos.

Finalmente se tomaron los resultados obtenidos de los modelos y se realiza una comparación entre ambos para determinar que modelo logra mejores resultados de predicción. En nuestro caso se pudo determinar que el modelo realizado con Random Forest logra mejor accouracy que el modelo de Naive Bayes, por lo que las predicciones son mas acertadas. En este documento se describe el proceso realizado para lograr los resultados obtenidos.

## II. Marco Teórico

Generalmente, cuando un programa malicioso infecta un sistema, se suele hacer referencia a un "virus", sin embargo, puede tratarse de cualquier otro tipo de malware. De hecho, actualmente solo un bajo porcentaje de los códigos maliciosos que se desarrollan y propagan por Internet corresponde a los denominados virus.

En su lugar, otros tipos de malware han proliferado para afectar a los usuarios con nuevas y variadas técnicas de propagación e infección, mismos que pueden ser clasificados en función de sus características, propósitos o funcionalidades. Con el objetivo de tener más información al respecto, en los siguientes párrafos se describe como se preparó el dataset.

### *II-A. Comprensión de la Metodología*

Para evaluar y mejorar los métodos de predicción hacía los dos modelos ante las futuras amenazas de malware, se dividió el data entre 30 % de pruebas, 55 % de entrenamiento y 15 % de validación, con el fin de contener un alto porcentaje en las métricas.

### *II-B. Implementación de Tecnología*

#### *II-B1. Naive Bayes*

El clasificador Naive Bayes es un modelo de clasificación de aprendizaje de maquina probabilistico que se basa en el teorema de Bayes.

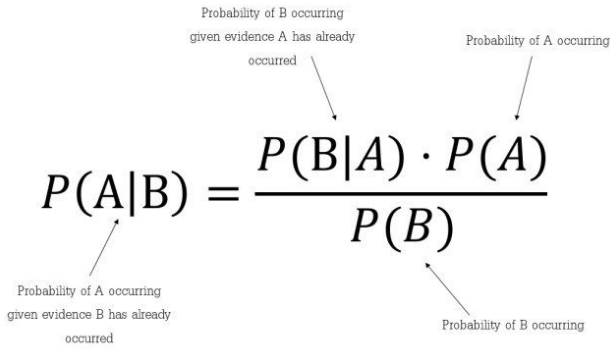


Fig. 1. Formula del Clasificador Naive Bayes

Este modelo es fácil de construir y especialmente útil para conjuntos de datos muy grandes. Además de su simplicidad,

se sabe que Naive Bayes supera incluso a los métodos de clasificación más sofisticados, y es ideal para los datos obtenidos para este proyecto.

### *II-B2. Random Forest*

El modelo de Random Forest es un método de aprendizaje automático supervisado construido a partir de técnicas de árboles de decisión. Contiene de varios árboles de decisión, creando diferentes nodos de manera que entrena mediante la agregación de bolsas o bootstrap.

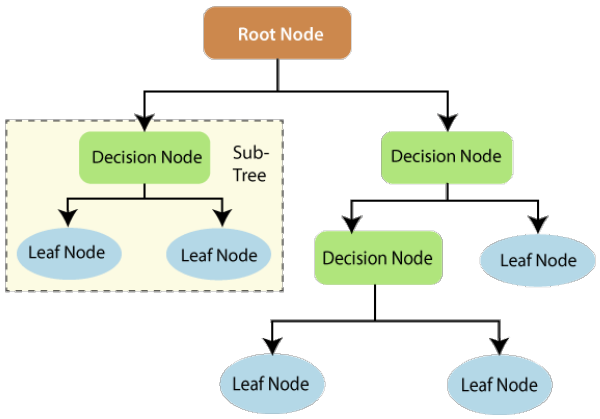


Fig. 2. Diagrama de Random Forest

Realiza las predicciones tomando la media de los resultados de los distintos árboles. El aumento del número de árboles mejora la precisión del resultado.

## III. Metodología

A continuación se muestran las fases del proyecto con una explicación del objetivo y las tareas que se realizaron en cada una de las mismas.

### *III-A. EDA*

Primero obtuvimos una vista general y clara de la información que íbamos a utilizar para crear nuestros modelos, comprobar si todas las columnas iban a ser necesarias, encontrar valores nulos, además de conocer la distribución de las variables iniciales.

### *III-B. Preprocesamiento*

Antes de generar nuestros modelos se realizó un preprocesamiento de información obtenida desde un archivo .csv con 8,921,483 observaciones. Sin embargo, al haber explorado el archivo se eliminaron las columnas que no iban servir para generar modelo, porque tenían valores únicos, o en su mayoría tenían solo valores NULL, o el valor de la variable no aportaba en un modelo de predicción. Las variables eliminadas fueron:

- MachineIdentifier
- EngineVersion
- AppVersion
- AvSigVersion
- OsVer
- OsBuildLab
- PuaMode
- SmartScreen
- SmartScreen
- Census_ProcessorClass
- Census_ChassisTypeName
- Census_InternalBatteryType
- Census_OSVersion
- Census_OSBranch
- Census_OSEdition
- Census_OSSkuName
- Census_OSInstallTypeName

Dentro del dataset existen variables categóricas que fueron necesarias convertirlas a numéricas para que fueran más fáciles de manipular al momento de realizar los modelos.
Las variables categorías que se convirtieron a numéricas fueron:

- ProductName
- Platform
- Processor
- OsPlatformSubRelease
- SkuEdition
- Census_MDC2FormFactor
- Census_DeviceFamily
- Census_PrimaryDiskTypeName
- Census_PowerPlatformRoleName
- Census_OSArchitecture
- Census_OSWUAutoUpdateOptionsName
- Census_GenuineStateName
- Census_ActivationChannel
- Census_FlightRing

Como parte del procesamiento de información se intentaron eliminar las filas que contenían valores NULL, ya que esto puede causar problemas al momento de generar los modelos, de este proceso se encontró que varías variables categóricas tenían en su mayoría valores NULL, por lo que también se eliminaron estás columnas ya que no aportan mucha información para la predicción de los modelos.
Las filas eliminadas que en su mayoría tienen valores NULL son:

- DefaultBrowsersIdentifier
- CityIdentifier
- OrganizationIdentifier
- SMode
- Firewall
- Census_OEMNameIdentifier
- Census_OEMModelIdentifier
- Census_TotalPhysicalRAM
- Census_InternalBatteryNumberOfCharges
- Census_IsFlightingInternal
- Census_IsFlightsDisabled
- Census_ThresholdOptIn

- Census_FirmwareManufacturerIdentifier
- Census_FirmwareVersionIdentifier
- Census_IsWIMBootEnabled
- Wdft_IsGamer
- Wdft_RegionIdentifier

Finalmente, pudimos obtener un dataset limpio con 8,588,801 observaciones con 50 variables, al cual se realizó un proceso de ajuste del tipo de dato para cada columna para reducir la memoria utilizada al momento de generar nuestros modelos.

### *III-C. Naive Bayes*

Luego del preprocesamiento se pudo comenzar a crear y entrenar nuestros modelos basado en el modelo de Naive Bayes. Se escogió realizar un modelo de Naive Bayes Gausiano ya que existian variables predictoras continuas y por lo tanto esta opción es la indicada para medir este tipo de variables. Luego de esto se procedio a la separación de los datos para entrenamento validación y pruebas y por último se analizaron las metricas obtenidas. Este analisis se puede observar en la siguiente sección.

### *III-D. Random Forest*

El segundo modelo que seleccionamos fue Random Forest, este modelo está compuesto de árboles de decisión con muestras al azar para luego obtener las predicciones de cada árbol y selecciona la mejor solución por medio de votación, dado que son utilizados para realizar predicciones y el dataset que provisto para el análisis tiene variables independientes, utilizar el modelo fue de gran ayuda para crear un modelo de predicción, que era el objetivo del proyecto. Luego de procesar y separar la información, creamos el modelo y se analizaron los resultados obtenidos.

## IV. Resultados

### *IV-A. Naive Bayes*

Los resultados obtenidos en con el Modelo de Naive Bayes se pueden observar en las siguientes figuras y al evaluarlo con evaluación cruzada, se obtuvo una precision del 50 % con una desviación estandar de -0.146

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.48 | 0.00 | 0.00 | 1289369 |
| 1 | 0.50 | 1.00 | 0.67 | 1287272 |
| accuracy | | | 0.50 | 2576641 |
| macro avg | 0.49 | 0.50 | 0.33 | 2576641 |
| weighted avg | 0.49 | 0.50 | 0.33 | 2576641 |

Fig. 3. Metricas del Modelo Naive Bayes

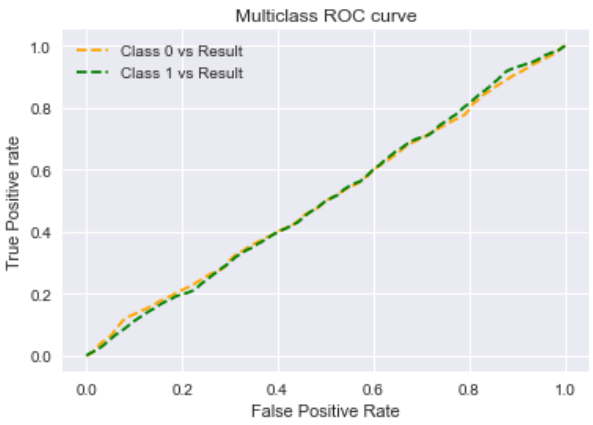


Fig. 4. Curva Roc Naive Bayes

Los resultados de las métricas podemos observar que tiene una precisión baja, causando incertidumbres respecto a las clasificaciones del malware, con respecto al objetivo que se había planteado en los modelos. Una de las razones que pudo haber afectado el rendimiento de este modelo fue que el conjunto de datos de prueba asume que todas las características son independientes. Aunque en teoría puede ayudar bastante en otros modelos como el Random Forest, las posibilidades que todas las características sean independientes son bajas.

### *IV-B. Random Forest*

Con el modelo de Random Forest se logra un acurracy de 0.60

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.60 | 0.60 | 0.60 | 1289369 |
| 1 | 0.60 | 0.60 | 0.60 | 1287272 |
| accuracy | | | 0.60 | 2576641 |
| macro avg | 0.60 | 0.60 | 0.60 | 2576641 |
| weighted avg | 0.60 | 0.60 | 0.60 | 2576641 |

Fig. 5. Metricas del Modelo Random Forest

Como podemos observar en las metricas, con el modelo se logra una precisión de 0.60

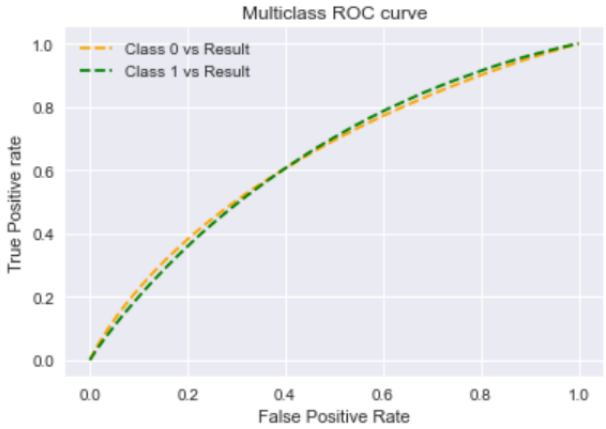


Fig. 6. Curva Roc Random Forest

La validación obtenida con Kfold para el modelo de Random Forest es: Accuracy: 0.926 (0.012), obteniendo una muy buena validación para el modelo generado.

Como se puede observar en las metricas generadas, la presición para ambos modelos es muy baja, sin embargo, podemos notar que el modelo de Random Forest logró una mejor predicción para determinar si la computadora sería infectada o no, ya que tienen mejores metricas comparada con el modelo de Naive Bayes.

Debido a que se creó una función en el programa donde se reduce el uso de memoria para esta cantidad de datos, se sugiere combinar este metódo conjunto con el PCA(Principal Component Analysis). Aplicando este modelo, se aumentaría las métricas de nuestros modelos, debido a que es un método de reducción de dimensionalidad que permite simplificar la complejidad de espacios con múltiples dimensiones a la vez que conserva su información.

## V. Conclusiones

- El modelo de Random Forest tuvo mejores métricas respecto al modelo de Naive Bayes para el conjunto de datos que se trabajó, por lo que podemos afirmar que el modelo de Random Forest podría predecir mejor si una computadora será infectada o no, en base a sus características.
- En ambos modelos se obtuvieron métricas muy bajas como para poder establecer que un modelo es realmente eficiente prediciendo que computadoras serán infectadas, esto se debe a que a lo mejor no se seleccionaron correctamente las variables a utilizar y que es necesario se apliquen otras técnicas para preprocesar la información con el fin de obtener modelos más confiables.
- A pesar de tener métricas bajas, se encontró una manera de mejoraría hacia ambos modelos utilizando el PCA, ya que çondensa"la información aportada por múltiples variables en solo unas pocas componentes, aliviando el peso del conjunto de datos.

## Referencias


[1] O. Mbaabu(2020). Ïntroduction to Random Forest in Machine Learning". Extraído de: https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/

[2] Pykes, K (2020) .Oversampling and Undersampling.Extraído de: https://towardsdatascience.com/oversampling-and-undersampling-5e2bbaf56dcf

[3] Grandhi, R (2018) "Naive BayesClassifier.extraído de: https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c

[4] Navlani, A. (2018). Understanding Random Forests Classifiers in Python Tutorial. https://www.datacamp.com/community/tutorials/random-forests-classifier-python

[5] Browniee, J. (2020) ROC Curves and Precision-Recall Curves for Imbalanced Classification https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-imbalanced-classification