1. First of all, we´ll import the libreries

```
import sys
import os

import numpy as np # Arrays
import pandas as pd # Series and Dataframes

import matplotlib.pyplot as plt
import seaborn as sns # Advanced Plotting
from google.colab import files


pd.options.display.max_rows = 100
plt.style.use('ggplot') # Beautify Plots

import random
import json
from pandas.io.json import json_normalize
from scipy import stats
from datetime import datetime
from sklearn import model_selection
import xgboost as xgb
```

Reading the data

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
df_virus=pd.read_csv("/content/drive/MyDrive/Colab Notebooks/Entregable 2/sample_mmp.csv")
```

```
<ipython-input-3-b7da9c851d4c>:1: DtypeWarning: Columns (29,42) have mixed types. Specify dtype option on import or set low_memory=
  df_virus=pd.read_csv("/content/drive/MyDrive/Colab Notebooks/Entregable 2/sample_mmp.csv")
```

2. Data Understanding

2.1. Univariant analysis of the data

2.1.1. Dataset´s size

```
df_virus.info(verbose = False)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500000 entries, 0 to 499999
Columns: 84 entries, Unnamed: 0 to HasDetections
dtypes: float64(36), int64(18), object(30)
memory usage: 320.4+ MB
```

MLC 2.1.2 Direct visualization of the data

```
df_virus.columns
```

```
Index(['Unnamed: 0', 'MachineIdentifier', 'ProductName', 'EngineVersion',
       'AppVersion', 'AvSigVersion', 'IsBeta', 'RtpStateBitfield',
       'IsSxsPassiveMode', 'DefaultBrowsersIdentifier',
       'AVProductStatesIdentifier', 'AVProductsInstalled', 'AVProductsEnabled',
       'HasTpm', 'CountryIdentifier', 'CityIdentifier',
       'OrganizationIdentifier', 'GeoNameIdentifier',
       'LocaleEnglishNameIdentifier', 'Platform', 'Processor', 'OsVer',
       'OsBuild', 'OsSuite', 'OsPlatformSubRelease', 'OsBuildLab',
       'SkuEdition', 'IsProtected', 'AutoSampleOptIn', 'PuaMode', 'SMode',
       'IeVerIdentifier', 'SmartScreen', 'Firewall', 'UacLuaenable',
       'Census_MDC2FormFactor', 'Census_DeviceFamily',
       'Census_OEMNameIdentifier', 'Census_OEMModelIdentifier',
       'Census_ProcessorCoreCount', 'Census_ProcessorManufacturerIdentifier',
       'Census_ProcessorModelIdentifier', 'Census_ProcessorClass',
       'Census_PrimaryDiskTotalCapacity', 'Census_PrimaryDiskTypeName',
       'Census_SystemVolumeTotalCapacity', 'Census_HasOpticalDiskDrive',
       'Census_TotalPhysicalRAM', 'Census_ChassisTypeName',
       'Census_InternalPrimaryDiagonalDisplaySizeInInches',
```

```
       'Census_InternalPrimaryDisplayResolutionHorizontal',
       'Census_InternalPrimaryDisplayResolutionVertical',
       'Census_PowerPlatformRoleName', 'Census_InternalBatteryType',
       'Census_InternalBatteryNumberOfCharges', 'Census_OSVersion',
       'Census_OSArchitecture', 'Census_OSBranch', 'Census_OSBuildNumber',
       'Census_OSBuildRevision', 'Census_OSEdition', 'Census_OSSkuName',
       'Census_OSInstallTypeName', 'Census_OSInstallLanguageIdentifier',
       'Census_OSUILocaleIdentifier', 'Census_OSWUAutoUpdateOptionsName',
       'Census_IsPortableOperatingSystem', 'Census_GenuineStateName',
       'Census_ActivationChannel', 'Census_IsFlightingInternal',
       'Census_IsFlightsDisabled', 'Census_FlightRing',
       'Census_ThresholdOptIn', 'Census_FirmwareManufacturerIdentifier',
       'Census_FirmwareVersionIdentifier', 'Census_IsSecureBootEnabled',
       'Census_IsWIMBootEnabled', 'Census_IsVirtualDevice',
       'Census_IsTouchEnabled', 'Census_IsPenCapable',
       'Census_IsAlwaysOnAlwaysConnectedCapable', 'Wdft_IsGamer',
       'Wdft_RegionIdentifier', 'HasDetections'],
      dtype='object')
```

```
df_virus.head(3).T
```

	0	1	
Unnamed: 0	8427007	8829090	
MachineIdentifier	f1cd864e97bae82bdf96523e1a539121	fd5ba6f5b75325ec0423a6c67cc75942	4e62
ProductName	win8defender	win8defender	
EngineVersion	1.1.15100.1	1.1.15100.1	
AppVersion	4.18.1807.18075	4.18.1807.18075	
AvSigVersion	1.273.1234.0	1.273.1282.0	
IsBeta	0	0	
RtpStateBitfield	7.0	7.0	
IsSxsPassiveMode	0	0	
DefaultBrowsersIdentifier	NaN	NaN	
AVProductStatesIdentifier	53447.0	53447.0	
AVProductsInstalled	1.0	1.0	
AVProductsEnabled	1.0	1.0	
HasTpm	1	1	
CountryIdentifier	8	129	
CityIdentifier	85219.0	54198.0	
OrganizationIdentifier	NaN	NaN	
GeoNameIdentifier	205.0	126.0	
LocaleEnglishNameIdentifier	172	124	
Platform	windows10	windows10	
Processor	x64	x64	
OsVer	10.0.0.0	10.0.0.0	
OsBuild	17134	17134	
OsSuite	256	256	
OsPlatformSubRelease	rs4	rs4	
OsBuildLab	17134.1.amd64fre.rs4_release.180410-1804	17134.1.amd64fre.rs4_release.180410-1804	10586.1176.an
SkuEdition	Pro	Pro	
IsProtected	1.0	1.0	
AutoSampleOptIn	0	0	
PuaMode	NaN	NaN	
SMode	0.0	0.0	
IeVerIdentifier	137.0	137.0	
SmartScreen	RequireAdmin	RequireAdmin	
Firewall	1.0	1.0	
UacLuaenable	1.0	1.0	
Census_MDC2FormFactor	Desktop	Notebook	
Census_DeviceFamily	Windows.Desktop	Windows.Desktop	
Census_OEMNameIdentifier	1443.0	2102.0	
Census_OEMModelIdentifier	275891.0	248850.0	
Census_ProcessorCoreCount	4.0	4.0	
Census_ProcessorManufacturerIdentifier	5.0	5.0	
Census_ProcessorModelIdentifier	2273.0	2660.0	
Census_ProcessorClass	NaN	NaN	
Census_PrimaryDiskTotalCapacity	953869.0	476940.0	
Census_PrimaryDiskTypeName	HDD	HDD	
Census_SystemVolumeTotalCapacity	952838.0	457600.0	

- MachineIdentifier - Individual machine ID
- ProductName - Defender state information e.g. win8defender

- EngineVersion - Defender state information e.g. 1.1.12603.0
- AppVersion - Defender state information e.g. 4.9.10586.0
- AvSigVersion - Defender state information e.g. 1.217.1014.0
- IsBeta - Defender state information e.g. false
- RtpStateBitfield - NA
- IsSxsPassiveMode - NA
- DefaultBrowsersIdentifier - ID for the machine's default browser
- AVProductStatesIdentifier - ID for the specific configuration of a user's antivirus software
- AVProductsInstalled - NA
- AVProductsEnabled - NA
- HasTpm - True if machine has tpm
- CountryIdentifier - ID for the country the machine is located in
- CityIdentifier - ID for the city the machine is located in
- OrganizationIdentifier - ID for the organization the machine belongs in, organization ID is mapped to both specific companies and broad industries
- GeoNameIdentifier - ID for the geographic region a machine is located in
- LocaleEnglishNameIdentifier - English name of Locale ID of the current user
- Platform - Calculates platform name (of OS related properties and processor property)
- Processor - This is the process architecture of the installed operating system
- OsVer - Version of the current operating system
- OsBuild - Build of the current operating system
- OsSuite - Product suite mask for the current operating system.
- OsPlatformSubRelease - Returns the OS Platform sub-release (Windows Vista, Windows 7, Windows 8, TH1, TH2)
- OsBuildLab - Build lab that generated the current OS. Example: 9600.17630.amd64fre.winblue_r7.150109-2022
- SkuEdition - The goal of this feature is to use the Product Type defined in the MSDN to map to a 'SKU-Edition' name that is useful in population reporting. The valid Product Type are defined in %sdxroot%\data\windowseditions.xml. This API has been used since Vista and Server 2008, so there are many Product Types that do not apply to Windows 10. The 'SKU-Edition' is a string value that is in one of three classes of results. The design must hand each class.
- IsProtected - This is a calculated field derived from the Spynet Report's AV Products field. Returns: a. TRUE if there is at least one active and up-to-date antivirus product running on this machine. b. FALSE if there is no active AV product on this machine, or if the AV is active, but is not receiving the latest updates. c. null if there are no Anti Virus Products in the report. Returns: Whether a machine is protected.
- AutoSampleOptIn - This is the SubmitSamplesConsent value passed in from the service, available on CAMP 9+
- PuaMode - Pua Enabled mode from the service
- SMode - This field is set to true when the device is known to be in 'S Mode', as in, Windows 10 S mode, where only Microsoft Store apps can be installed
- IeVerIdentifier - NA SmartScreen - This is the SmartScreen enabled string value from registry. This is obtained by checking in order, HKLM\SOFTWARE\Policies\Microsoft\Windows\System\SmartScreenEnabled and HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\SmartScreenEnabled. If the value exists but is blank, the value "ExistsNotSet" is sent in telemetry.
- Firewall - This attribute is true (1) for Windows 8.1 and above if windows firewall is enabled, as reported by the service.
- UacLuaenable - This attribute reports whether or not the "administrator in Admin Approval Mode" user type is disabled or enabled in UAC. The value reported is obtained by reading the regkey HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System\EnableLUA.
- Census_MDC2FormFactor - A grouping based on a combination of Device
- Census level hardware characteristics. The logic used to define Form Factor is rooted in business and industry standards and aligns with how people think about their device. (Examples: Smartphone, Small Tablet, All in One, Convertible...)
- Census_DeviceFamily - AKA DeviceClass. Indicates the type of device that an edition of the OS is intended for. Example values: Windows.Desktop, Windows.Mobile, and iOS.Phone
- Census_OEMNameIdentifier - NA
- Census_OEMModelIdentifier - NA
- Census_ProcessorCoreCount - Number of logical cores in the processor
- Census_ProcessorManufacturerIdentifier - NA
- Census_ProcessorModelIdentifier - NA
- Census_ProcessorClass - A classification of processors into high/medium/low. Initially used for Pricing Level SKU. No longer maintained and updated
- Census_PrimaryDiskTotalCapacity - Amount of disk space on primary disk of the machine in MB
- Census_PrimaryDiskTypeName - Friendly name of Primary Disk Type - HDD or SSD
- Census_SystemVolumeTotalCapacity - The size of the partition that the System volume is installed on in MB
- Census_HasOpticalDiskDrive - True indicates that the machine has an optical disk drive (CD/DVD)
- Census_TotalPhysicalRAM - Retrieves the physical RAM in MB
- Census_ChassisTypeName - Retrieves a numeric representation of what type of chassis the machine has. A value of 0 means xx

- Census_InternalPrimaryDiagonalDisplaySizeInInches - Retrieves the physical diagonal length in inches of the primary display
- Census_InternalPrimaryDisplayResolutionHorizontal - Retrieves the number of pixels in the horizontal direction of the internal display.
- Census_InternalPrimaryDisplayResolutionVertical - Retrieves the number of pixels in the vertical direction of the internal display
- Census_PowerPlatformRoleName - Indicates the OEM preferred power management profile. This value helps identify the basic form factor of the device
- Census_InternalBatteryType - NA
- Census_InternalBatteryNumberOfCharges - NA
- Census_OSVersion - Numeric OS version Example - 10.0.10130.0
- Census_OSArchitecture - Architecture on which the OS is based. Derived from OSVersionFull. Example - amd64
- Census_OSBranch - Branch of the OS extracted from the OsVersionFull. Example - OsBranch = fbl_partner_eeap where OsVersion = 6.4.9813.0.amd64fre.fbl_partner_eeap.140810-0005
- Census_OSBuildNumber - OS Build number extracted from the OsVersionFull. Example - OsBuildNumber = 10512 or 10240
- Census_OSBuildRevision - OS Build revision extracted from the OsVersionFull. Example - OsBuildRevision = 1000 or 16458
- Census_OSEdition - Edition of the current OS. Sourced from HKLM\Software\Microsoft\Windows NT\CurrentVersion@EditionID in registry. Example: Enterprise
- Census_OSSkuName - OS edition friendly name (currently Windows only)
- Census_OSInstallTypeName - Friendly description of what install was used on the machine i.e. clean
- Census_OSInstallLanguageIdentifier - NA
- Census_OSUILocaleIdentifier - NA
- Census_OSWUAutoUpdateOptionsName - Friendly name of the WindowsUpdate auto-update settings on the machine.
- Census_IsPortableOperatingSystem - Indicates whether OS is booted up and running via Windows-To-Go on a USB stick.
- Census_GenuineStateName - Friendly name of OSGenuineStateID. 0 = Genuine
- Census_ActivationChannel - Retail license key or Volume license key for a machine.
- Census_IsFlightingInternal - NA
- Census_IsFlightsDisabled - Indicates if the machine is participating in flighting.
- Census_FlightRing - The ring that the device user would like to receive flights for. This might be different from the ring of the OS which is currently installed if the user changes the ring after getting a flight from a different ring.
- Census_ThresholdOptIn - NA
- Census_FirmwareManufacturerIdentifier - NA
- Census_FirmwareVersionIdentifier - NA
- Census_IsSecureBootEnabled - Indicates if Secure Boot mode is enabled.
- Census_IsWIMBootEnabled - NA
- Census_IsVirtualDevice - Identifies a Virtual Machine (machine learning model)
- Census_IsTouchEnabled - Is this a touch device ?
- Census_IsPenCapable - Is the device capable of pen input ?
- Census_IsAlwaysOnAlwaysConnectedCapable - Retreives information about whether the battery enables the device to be AlwaysOnAlwaysConnected
- Wdft_IsGamer - Indicates whether the device is a gamer device or not based on its hardware combination.
- Wdft_RegionIdentifier - NA

MLC 2.1.3 Types of atributes

```
df_virus.info(verbose = True)
```

```
 48  Census_ChassisTypeName                                  499963 non-null  object
 49  Census_InternalPrimaryDiagonalDisplaySizeInInches       497346 non-null  float64
 50  Census_InternalPrimaryDisplayResolutionHorizontal       497350 non-null  float64
 51  Census_InternalPrimaryDisplayResolutionVertical         497350 non-null  float64
 52  Census_PowerPlatformRoleName                            499998 non-null  object
 53  Census_InternalBatteryType                              144397 non-null  object
 54  Census_InternalBatteryNumberOfCharges                   484962 non-null  float64
 55  Census_OSVersion                                        500000 non-null  object
 56  Census_OSArchitecture                                   500000 non-null  object
 57  Census_OSBranch                                         500000 non-null  object
 58  Census_OSBuildNumber                                    500000 non-null  int64
 59  Census_OSBuildRevision                                  500000 non-null  int64
 60  Census_OSEdition                                        500000 non-null  object
 61  Census_OSSkuName                                        500000 non-null  object
 62  Census_OSInstallTypeName                                500000 non-null  object
 63  Census_OSInstallLanguageIdentifier                      496668 non-null  float64
 64  Census_OSUILocaleIdentifier                             500000 non-null  int64
 65  Census_OSWUAutoUpdateOptionsName                        500000 non-null  object
 66  Census_IsPortableOperatingSystem                        500000 non-null  int64
 67  Census_GenuineStateName                                 500000 non-null  object
 68  Census_ActivationChannel                                500000 non-null  object
 69  Census_IsFlightingInternal                              84775 non-null   float64
 70  Census_IsFlightsDisabled                                491067 non-null  float64
 71  Census_FlightRing                                       500000 non-null  object
 72  Census_ThresholdOptIn                                   181896 non-null  float64
 73  Census_FirmwareManufacturerIdentifier                   489651 non-null  float64
 74  Census_FirmwareVersionIdentifier                        490939 non-null  float64
 75  Census_IsSecureBootEnabled                              500000 non-null  int64
 76  Census_IsWIMBootEnabled                                 182334 non-null  float64
 77  Census_IsVirtualDevice                                  499099 non-null  float64
 78  Census_IsTouchEnabled                                   500000 non-null  int64
 79  Census_IsPenCapable                                     500000 non-null  int64
 80  Census_IsAlwaysOnAlwaysConnectedCapable                 495960 non-null  float64
```

```
def separar_tipo_atributos():
    boolean = [col for col in df_virus.columns if (df_virus[col].nunique() == 2) & (col not in 'HasDetections')]
    categoricas = [col for col in df_virus.columns if (df_virus[col].dtypes == 'O') & (col not in 'HasDetections')]
    numericas = [col for col in df_virus.columns if (col not in categoricas) & (col not in boolean) & (col not in 'HasDetections')]
    return boolean, numericas, categoricas

columnas_booleanas, columnas_numericas, columnas_categoricas = separar_tipo_atributos()
```

MLC 2.1.4 Statistics & descriptive

```
df_virus.describe(include = ['object']).T
```

	count	unique	top	freq
MachineIdentifier	500000	500000	f1cd864e97bae82bdf96523e1a539121	1
ProductName	500000	3	win8defender	494604
EngineVersion	500000	53	1.1.15200.1	216491
AppVersion	500000	95	4.18.1807.18075	288809
AvSigVersion	500000	6455	1.273.1420.0	5771
Platform	500000	4	windows10	483048
Processor	500000	3	x64	454423
OsVer	500000	21	10.0.0.0	483830

```
df_virus.describe(exclude = ['object']).T
```

	count	mean	std	min	25%	50%	75%
Unnamed: 0	500000.0	4.458888e+06	2.575619e+06	2.0	2227692.50	4461367.5	6.690936e+06
IsBeta	500000.0	2.000000e-06	1.414214e-03	0.0	0.00	0.0	0.000000e+00
RtpStateBitfield	498168.0	6.846207e+00	1.023049e+00	0.0	7.00	7.0	7.000000e+00
IsSxsPassiveMode	500000.0	1.724200e-02	1.301720e-01	0.0	0.00	0.0	0.000000e+00
DefaultBrowsersIdentifier	24061.0	1.652825e+03	1.004754e+03	1.0	788.00	1632.0	2.381000e+03
AVProductStatesIdentifier	498062.0	4.785091e+04	1.402309e+04	3.0	49480.00	53447.0	5.344700e+04
AVProductsInstalled	498062.0	1.326763e+00	5.229999e-01	1.0	1.00	1.0	2.000000e+00
AVProductsEnabled	498062.0	1.020714e+00	1.666080e-01	0.0	1.00	1.0	1.000000e+00
HasTpm	500000.0	9.878160e-01	1.097068e-01	0.0	1.00	1.0	1.000000e+00
CountryIdentifier	500000.0	1.080375e+02	6.306854e+01	1.0	51.00	97.0	1.620000e+02
CityIdentifier	481760.0	8.127165e+04	4.898513e+04	7.0	36825.00	82373.0	1.239395e+05
OrganizationIdentifier	345437.0	2.486994e+01	5.613712e+00	1.0	18.00	27.0	2.700000e+01
GeoNameIdentifier	499984.0	1.697304e+02	8.932517e+01	1.0	89.00	181.0	2.670000e+02
LocaleEnglishNameIdentifier	500000.0	1.226110e+02	6.930317e+01	1.0	74.00	88.0	1.820000e+02
OsBuild	500000.0	1.572693e+04	2.188646e+03	7600.0	15063.00	16299.0	1.713400e+04

MLC 2.1.5 Nulls

```
df_virus.select_dtypes(include = ['number']).isnull().sum()
```

```
Unnamed: 0                                        0
IsBeta                                            0
RtpStateBitfield                               1832
IsSxsPassiveMode                                  0
DefaultBrowsersIdentifier                    475939
AVProductStatesIdentifier                      1938
AVProductsInstalled                            1938
AVProductsEnabled                              1938
HasTpm                                            0
CountryIdentifier                                 0
CityIdentifier                                18240
OrganizationIdentifier                       154563
GeoNameIdentifier                                16
LocaleEnglishNameIdentifier                       0
OsBuild                                           0
OsSuite                                           0
IsProtected                                    1926
AutoSampleOptIn                                   0
SMode                                         29848
IeVerIdentifier                                3209
Firewall                                       5162
UacLuaenable                                    623
Census_OEMNameIdentifier                       5381
Census_OEMModelIdentifier                      5764
Census_ProcessorCoreCount                      2347
Census_ProcessorManufacturerIdentifier         2347
Census_ProcessorModelIdentifier                2349
Census_PrimaryDiskTotalCapacity                2976
Census_SystemVolumeTotalCapacity               2976
Census_HasOpticalDiskDrive                        0
Census_TotalPhysicalRAM                        4556
Census_InternalPrimaryDiagonalDisplaySizeInInches      2654
Census_InternalPrimaryDisplayResolutionHorizontal      2650
Census_InternalPrimaryDisplayResolutionVertical        2650
Census_InternalBatteryNumberOfCharges         15038
Census_OSBuildNumber                              0
Census_OSBuildRevision                            0
Census_OSInstallLanguageIdentifier             3332
Census_OSUILocaleIdentifier                       0
Census_IsPortableOperatingSystem                  0
Census_IsFlightingInternal                   415225
Census_IsFlightsDisabled                       8933
Census_ThresholdOptIn                        318104
Census_FirmwareManufacturerIdentifier         10349
Census_FirmwareVersionIdentifier               9061
Census_IsSecureBootEnabled                        0
Census_IsWIMBootEnabled                      317666
Census_IsVirtualDevice                          901
Census_IsTouchEnabled                             0
Census_IsPenCapable                               0
Census_IsAlwaysOnAlwaysConnectedCapable        4040
Wdft_IsGamer                                  16950
Wdft_RegionIdentifier                         16950
```

```
HasDetections                                     0
dtype: int64
```

```
Census_IsAlwaysOnAlwaysConnectedCapable  495960.0  5.724454e-02  2.323095e-01  0.0  0.00  0.0  0.000000e+00
```

```
df_virus.select_dtypes(include = ['object']).isnull().sum()
```

```
MachineIdentifier                       0
ProductName                             0
EngineVersion                           0
AppVersion                              0
AvSigVersion                            0
Platform                                0
Processor                               0
OsVer                                   0
OsPlatformSubRelease                    0
OsBuildLab                              1
SkuEdition                              0
PuaMode                            499874
SmartScreen                        178596
Census_MDC2FormFactor                   0
Census_DeviceFamily                     0
Census_ProcessorClass              497918
Census_PrimaryDiskTypeName            709
Census_ChassisTypeName                 37
Census_PowerPlatformRoleName            2
Census_InternalBatteryType         355603
Census_OSVersion                        0
Census_OSArchitecture                   0
Census_OSBranch                         0
Census_OSEdition                        0
Census_OSSkuName                        0
Census_OSInstallTypeName                0
Census_OSWUAutoUpdateOptionsName        0
Census_GenuineStateName                 0
Census_ActivationChannel                0
Census_FlightRing                       0
dtype: int64
```

MLC 2.1.6 TARGET analysis

```
TARGET = 'HasDetections'
```

```
df_virus[TARGET].value_counts()
```

```
0    250047
1    249953
Name: HasDetections, dtype: int64
```

```
df_virus[TARGET].mean()
```

```
0.499906
```

The mean is 49,99%, so we can see that half of the machines are detected by Malware

```
df_virus2 = df_virus.copy()
```

2.1.7 - 2.1.9: Variables distribution

```
# Checking the number of the columns considering type
```

```
len(columnas_booleanas)
```

```
18
```

```
len(columnas_categoricas)
```

```
30
```

```
len(columnas_numericas)
```

```
35
```

```
df_virus2.head(2).T
```

	0	1
Unnamed: 0	8427007	8829090
MachineIdentifier	f1cd864e97bae82bdf96523e1a539121	fd5ba6f5b75325ec0423a6c67cc75942
ProductName	win8defender	win8defender
EngineVersion	1.1.15100.1	1.1.15100.1
AppVersion	4.18.1807.18075	4.18.1807.18075
AvSigVersion	1.273.1234.0	1.273.1282.0
IsBeta	0	0
RtpStateBitfield	7.0	7.0
IsSxsPassiveMode	0	0
DefaultBrowsersIdentifier	NaN	NaN
AVProductStatesIdentifier	53447.0	53447.0
AVProductsInstalled	1.0	1.0
AVProductsEnabled	1.0	1.0
HasTpm	1	1
CountryIdentifier	8	129
CityIdentifier	85219.0	54198.0
OrganizationIdentifier	NaN	NaN
GeoNameIdentifier	205.0	126.0
LocaleEnglishNameIdentifier	172	124
Platform	windows10	windows10
Processor	x64	x64
OsVer	10.0.0.0	10.0.0.0
OsBuild	17134	17134
OsSuite	256	256
OsPlatformSubRelease	rs4	rs4
OsBuildLab	17134.1.amd64fre.rs4_release.180410-1804	17134.1.amd64fre.rs4_release.180410-1804
SkuEdition	Pro	Pro
IsProtected	1.0	1.0
AutoSampleOptIn	0	0
PuaMode	NaN	NaN
SMode	0.0	0.0
IeVerIdentifier	137.0	137.0
SmartScreen	RequireAdmin	RequireAdmin
Firewall	1.0	1.0
UacLuaenable	1.0	1.0
Census_MDC2FormFactor	Desktop	Notebook
Census_DeviceFamily	Windows.Desktop	Windows.Desktop
Census_OEMNameIdentifier	1443.0	2102.0
Census_OEMModelIdentifier	275891.0	248850.0
Census_ProcessorCoreCount	4.0	4.0
Census_ProcessorManufacturerIdentifier	5.0	5.0
Census_ProcessorModelIdentifier	2273.0	2660.0
Census_ProcessorClass	NaN	NaN
Census_PrimaryDiskTotalCapacity	953869.0	476940.0
Census_PrimaryDiskTypeName	HDD	HDD
Census_SystemVolumeTotalCapacity	952838.0	457600.0
Census_HasOpticalDiskDrive	0	0
Census_TotalPhysicalRAM	8192.0	8192.0
Census_ChassisTypeName	AllinOne	Notebook

Census_InternalPrimaryDiagonalDisplaySizeInInches	23.0	13.2
Census_InternalPrimaryDisplayResolutionHorizontal	1920.0	1280.0
Census_InternalPrimaryDisplayResolutionVertical	1080.0	720.0
Census_PowerPlatformRoleName	Desktop	Mobile
Census_InternalBatteryType	NaN	lion
Census_InternalBatteryNumberOfCharges	4294967295.0	0.0
Census_OSVersion	10.0.17134.165	10.0.17134.165
Census_OSArchitecture	amd64	amd64
Census_OSBranch	rs4_release	rs4_release
Census_OSBuildNumber	17134	17134
Census_OSBuildRevision	165	165
Census_OSEdition	Professional	Professional
Census_OSSkuName	PROFESSIONAL	PROFESSIONAL
Census_OSInstallTypeName	UUPUpgrade	UUPUpgrade
Census_OSInstallLanguageIdentifier	27.0	18.0
Census_OSUILocaleIdentifier	120	72
Census_OSWUAutoUpdateOptionsName	FullAuto	FullAuto
Census_IsPortableOperatingSystem	0	0
Census_GenuineStateName	IS_GENUINE	IS_GENUINE
Census_ActivationChannel	OEM:DM	OEM:DM
Census_IsFlightingInternal	NaN	NaN
Census_IsFlightsDisabled	0.0	0.0
Census_FlightRing	Retail	Retail
Census_ThresholdOptIn	NaN	0.0
Census_FirmwareManufacturerIdentifier	355.0	486.0
Census_FirmwareVersionIdentifier	19951.0	48753.0
Census_IsSecureBootEnabled	0	0
Census_IsWIMBootEnabled	NaN	0.0
Census_IsVirtualDevice	0.0	0.0
Census_IsTouchEnabled	0	0
Census_IsPenCapable	0	0
Census_IsAlwaysOnAlwaysConnectedCapable	0.0	0.0
Wdft_IsGamer	0.0	1.0
Wdft_RegionIdentifier	11.0	3.0

```
df_virus2_description = df_virus2.describe(include='all').T
```

```
df_virus2_description.to_csv('df_description.csv')
```

```
files.download('df_description.csv')
```

Categoricals

```
columnas_categoricas
```

```
['MachineIdentifier',
 'ProductName',
 'EngineVersion',
 'AppVersion',
 'AvSigVersion',
 'Platform',
 'Processor',
 'OsVer',
 'OsPlatformSubRelease',
 'OsBuildLab',
```

```
        'SkuEdition',
        'PuaMode',
        'SmartScreen',
        'Census_MDC2FormFactor',
        'Census_DeviceFamily',
        'Census_ProcessorClass',
        'Census_PrimaryDiskTypeName',
        'Census_ChassisTypeName',
        'Census_PowerPlatformRoleName',
        'Census_InternalBatteryType',
        'Census_OSVersion',
        'Census_OSArchitecture',
        'Census_OSBranch',
        'Census_OSEdition',
        'Census_OSSkuName',
        'Census_OSInstallTypeName',
        'Census_OSWUAutoUpdateOptionsName',
        'Census_GenuineStateName',
        'Census_ActivationChannel',
        'Census_FlightRing']
```

```
# We´ll define a function in order to group all variables with few records in "Others"

def setOthers(dataframe, column, num_values):
    top_categories = dataframe[column].value_counts().head(num_values)
    top_categories_list = top_categories.index.to_list()
    top_categories_list.append('Others')
    dataframe[column] = pd.Categorical(dataframe[column], categories=top_categories_list)
    return dataframe[column].fillna('Others')
```

MachineIdentifier

```
df_virus2['MachineIdentifier'].value_counts()
```

```
f1cd864e97bae82bdf96523e1a539121    1
74f5f637add24668804961d81cca7697    1
d1e0560ad7cd178b32599edb67142737    1
d03a6fb0852e2ad3380e689a80d23273    1
8b3a780e6980a83b1650e7fc873dcfd1    1
                                   ..
6154953dc05531cad9fb680cf39990b4    1
b345b8ccc72ba43e7417b2018fb61444    1
013b44e10f97fa8f71b6b76e24323d51    1
ce0e5ba490a2455a87d09184b9bbd578    1
7b45dc3537e17f16305c5983ca0a1cb9    1
Name: MachineIdentifier, Length: 500000, dtype: int64
```

```
# The ID doesn´t give us any kind of relevant information, so we´ll proceed to eliminate it.
```

```
df_virus2.drop('MachineIdentifier', axis = 1, inplace = True)
```

ProductName

```
df_virus2['ProductName'].isnull().sum()
```

```
0
```

```
df_virus2['ProductName'].value_counts()
```

```
win8defender     494604
mse                5395
mseprerelease         1
Name: ProductName, dtype: int64
```

```
# We´ll group "MSE" & "MSEPRERELEASE" due to are similars variables
```

```
df_virus2['ProductName'] = df_virus2['ProductName'].replace('mseprerelease', 'mse')
```

```
df_virus2['ProductName'].value_counts()
```

```
win8defender    494604
mse               5396
Name: ProductName, dtype: int64
```

```
# We´ll execute OHE at the end
```

EngineVersion

```
df_virus2['EngineVersion'].isnull().sum()
```

```
0
```

```
len(df_virus2['EngineVersion'].value_counts())
```

```
53
```

```
df_virus2['EngineVersion'].value_counts()
```

```
1.1.15200.1    216491
1.1.15100.1    205494
1.1.15000.2     14752
1.1.14901.4     11984
1.1.14600.4      9005
1.1.14800.3      7548
1.1.15300.6      6769
1.1.14104.0      5240
1.1.15300.5      3883
1.1.13504.0      3876
1.1.14700.5      2593
1.1.14500.5      2591
1.1.14405.2      1972
1.1.14306.0      1329
1.1.14202.0       840
1.1.14003.0       804
1.1.13303.0       516
1.1.13903.0       515
1.1.13804.0       513
1.1.13407.0       479
1.1.13601.0       420
1.1.12902.0       321
1.1.14305.0       270
1.1.13701.0       255
1.1.13704.0       255
1.1.13202.0       245
1.1.13103.0       235
1.1.13000.0       220
1.1.15000.1       138
1.1.12805.0       115
1.1.14901.3        71
1.1.14700.4        65
1.1.14700.3        61
1.1.14800.1        51
1.1.14500.2        19
1.1.14201.0        11
1.1.14303.0        10
1.1.12101.0         7
1.1.14103.0         5
1.1.12804.0         4
1.1.13802.0         4
1.1.13902.0         4
1.1.13803.0         3
1.1.12603.0         3
1.1.11701.0         3
1.1.14001.0         2
1.1.14002.0         2
1.1.14102.0         2
1.1.13406.0         1
1.1.12400.0         1
1.1.12706.0         1
1.1.13102.0         1
1.1.10401.0         1
Name: EngineVersion, dtype: int64
```

```
df_virus2['EngineVersion'].value_counts()
```

```
1.1.15200.1    216491
1.1.15100.1    205494
1.1.15000.2     14752
1.1.14901.4     11984
1.1.14600.4      9005
1.1.14800.3      7548
1.1.15300.6      6769
1.1.14104.0      5240
1.1.15300.5      3883
1.1.13504.0      3876
1.1.14700.5      2593
1.1.14500.5      2591
1.1.14405.2      1972
1.1.14306.0      1329
1.1.14202.0       840
1.1.14003.0       804
```

```
1.1.13303.0       516
1.1.13903.0       515
1.1.13804.0       513
1.1.13407.0       479
1.1.13601.0       420
1.1.12902.0       321
1.1.14305.0       270
1.1.13701.0       255
1.1.13704.0       255
1.1.13202.0       245
1.1.13103.0       235
1.1.13000.0       220
1.1.15000.1       138
1.1.12805.0       115
1.1.14901.3        71
1.1.14700.4        65
1.1.14700.3        61
1.1.14800.1        51
1.1.14500.2        19
1.1.14201.0        11
1.1.14303.0        10
1.1.12101.0         7
1.1.14103.0         5
1.1.12804.0         4
1.1.13802.0         4
1.1.13902.0         4
1.1.13803.0         3
1.1.12603.0         3
1.1.11701.0         3
1.1.14001.0         2
1.1.14002.0         2
1.1.14102.0         2
1.1.13406.0         1
1.1.12400.0         1
1.1.12706.0         1
1.1.13102.0         1
1.1.10401.0         1
Name: EngineVersion, dtype: int64
```

```
df_virus2['EngineVersion'] = df_virus2['EngineVersion'].str.slice(stop=6)
```

```
df_virus2['EngineVersion'].value_counts()
```

```
1.1.15    447527
1.1.14     44475
1.1.13      7542
1.1.12       452
1.1.11         3
1.1.10         1
Name: EngineVersion, dtype: int64
```

```
df_virus2['EngineVersion'] = df_virus2['EngineVersion'].replace('1.1.13', 'OlderVersion')
df_virus2['EngineVersion'] = df_virus2['EngineVersion'].replace('1.1.12', 'OlderVersion')
df_virus2['EngineVersion'] = df_virus2['EngineVersion'].replace('1.1.11', 'OlderVersion')
df_virus2['EngineVersion'] = df_virus2['EngineVersion'].replace('1.1.10', 'OlderVersion')
```

```
df_virus2['EngineVersion'].value_counts()
```

```
1.1.15          447527
1.1.14           44475
OlderVersion      7998
Name: EngineVersion, dtype: int64
```

```
df_virus2['EngineVersion'] = setOthers(df_virus2, 'EngineVersion', 2)
```

AppVersion

```
df_virus2['AppVersion'].isnull().sum()
```

```
0
```

```
len(df_virus2['AppVersion'].value_counts())
```

```
95
```

```
df_virus2['AppVersion'].value_counts()
```

```
4.18.1807.18075     288809
4.18.1806.18062      47641
4.12.16299.15        20197
```

```
4.10.209.0           15292
4.13.17134.1         14414
4.16.17656.18052     13185
4.13.17134.228       12729
4.9.10586.1106       11432
4.8.10240.17443      11385
4.14.17639.18041     10670
4.12.17007.18022      6470
4.9.10586.0           6213
4.11.15063.447        5100
4.10.14393.0          4399
4.11.15063.0          3874
4.12.17007.18011      3392
4.14.17613.18039      2991
4.8.10240.16384       2617
4.11.15063.1155       2477
4.10.14393.1794       2425
4.9.10586.494         1383
4.10.14393.1198       1271
4.9.10586.672         1198
4.13.17134.191        1182
4.12.17007.17123       775
4.9.10586.589          761
4.10.14393.1613        760
4.18.1809.2            733
4.13.17134.112         651
4.9.10586.1045         617
4.10.14393.1593        536
4.10.14393.1066        404
4.10.14393.953         403
4.9.218.0              334
4.9.10586.916          311
4.9.10586.965          266
4.9.10586.962          262
4.8.10240.17946        262
4.9.10586.839          248
4.9.10586.873          228
4.8.207.0              204
4.5.218.0              116
4.8.10240.17889        103
4.13.17134.319          97
4.8.204.0               96
4.8.10240.17202         94
4.8.10240.17914         84
4.8.10240.17071         77
4.8.10240.17394         74
4.14.17613.18038        66
4.10.14393.2273         54
4.8.10240.17319         53
4.8.10240.17146         50
4.10.205.0              49
4.8.10240.17354         48
4.7.205.0               46
4.6.305.0               44
4.18.1807.20063         39
```

```
df_virus2['AppVersion'] = df_virus2['AppVersion'].str.slice(stop=4)
```

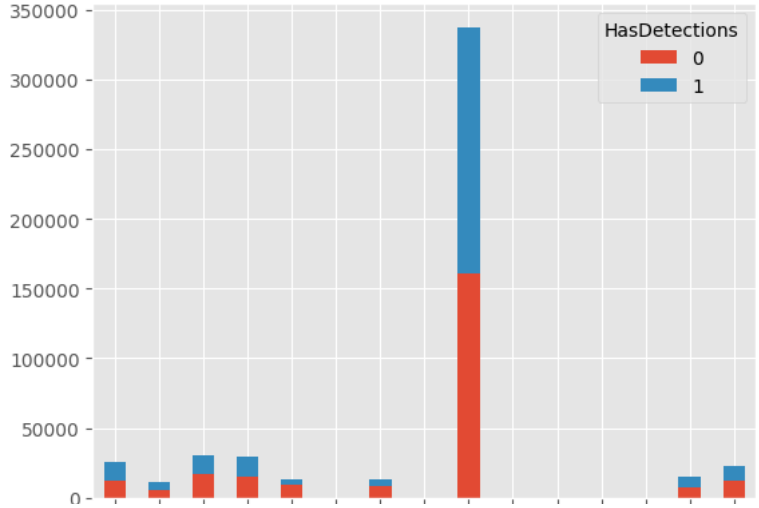
```
df_virus2['AppVersion'].value_counts()
```

```
4.18    337243
4.12     30836
4.13     29076
4.10     25633
4.9.     23256
4.8.     15309
4.14     13727
4.16     13189
4.11     11458
4.5.       126
4.7.        46
4.6.        44
4.4.        27
4.17        22
4.15         8
Name: AppVersion, dtype: int64
```

```
tabla_contingencia = pd.crosstab(df_virus2['AppVersion'], df_virus2['HasDetections'])
```

```
tabla_contingencia.plot(kind='bar', stacked=True)
```

```
<Axes: xlabel='AppVersion'>
```




```
# All versions that appear could have a media of 50% of detections, except 4.14 and 4.16 thay have 30%. These we´ll analyze in a deep ser
```

```
filtro_appversion_4_14 = df_virus2['AppVersion'] == '4.14'
```

```
df_virus2[filtro_appversion_4_14]['HasDetections'].mean()
```

```
0.3139068988125592
```

```
filtro_appversion_4_16 = df_virus2['AppVersion'] == '4.16'
```

```
df_virus2[filtro_appversion_4_16]['HasDetections'].mean()
```

```
0.3268632951702176
```

```
filtro_appversion_4_18 = df_virus2['AppVersion'] == '4.18'
```

```
df_virus2[filtro_appversion_4_18]['HasDetections'].mean()
```

```
0.5232695711994022
```

```
# All records that are not 4.18 (volume), 4.14 (target mean difference) or 4.16 (target mean difference) will be classified together
```

```
df_virus2['AppVersion'] = df_virus2['AppVersion'].replace('4.12', 'Others')
df_virus2['AppVersion'] = df_virus2['AppVersion'].replace('4.13', 'Others')
df_virus2['AppVersion'] = df_virus2['AppVersion'].replace('4.10', 'Others')
df_virus2['AppVersion'] = df_virus2['AppVersion'].replace('4.9.', 'Others')
df_virus2['AppVersion'] = df_virus2['AppVersion'].replace('4.8.', 'Others')
df_virus2['AppVersion'] = df_virus2['AppVersion'].replace('4.11', 'Others')
df_virus2['AppVersion'] = df_virus2['AppVersion'].replace('4.5.', 'Others')
df_virus2['AppVersion'] = df_virus2['AppVersion'].replace('4.7.', 'Others')
df_virus2['AppVersion'] = df_virus2['AppVersion'].replace('4.6.', 'Others')
df_virus2['AppVersion'] = df_virus2['AppVersion'].replace('4.4.', 'Others')
df_virus2['AppVersion'] = df_virus2['AppVersion'].replace('4.17', 'Others')
df_virus2['AppVersion'] = df_virus2['AppVersion'].replace('4.15', 'Others')
```

```
df_virus2['AppVersion'].value_counts()
```

```
4.18      337243
Others    135841
4.14       13727
4.16       13189
Name: AppVersion, dtype: int64
```

```
# We´ll execute OHE once we analyze all the variables
```

AvSigVersion

```
df_virus2['AvSigVersion'].isnull().sum()
```

```
0
```

```
len(df_virus2['AvSigVersion'].value_counts())
```

```
6455
```

```
df_virus2['AvSigVersion'].value_counts().head(50)
```

```
1.273.1420.0    5771
1.263.48.0      5537
1.275.1140.0    5317
1.275.727.0     5214
1.273.371.0     4799
1.273.1826.0    4744
1.275.1244.0    4487
1.251.42.0      4258
1.275.1209.0    3787
1.273.810.0     3708
1.237.0.0       3534
1.273.1749.0    3508
1.273.1379.0    2789
1.273.1005.0    2747
1.273.894.0     2540
1.273.781.0     2409
1.273.461.0     2332
1.273.337.0     2219
1.273.1527.0    2218
1.275.948.0     2170
1.275.1293.0    2086
1.273.1167.0    2081
1.273.717.0     2060
1.273.950.0     1979
1.275.1487.0    1923
1.275.1025.0    1916
1.273.1034.0    1872
1.275.511.0     1807
1.273.1112.0    1796
1.273.1311.0    1769
1.275.981.0     1738
1.275.1011.0    1732
1.273.1795.0    1730
1.275.1669.0    1708
1.275.974.0     1683
1.275.112.0     1670
1.273.1056.0    1654
1.275.263.0     1633
1.273.665.0     1628
1.275.1086.0    1615
1.273.1282.0    1605
1.273.1574.0    1551
1.273.1073.0    1523
1.273.1668.0    1497
1.273.933.0     1484
1.275.1458.0    1468
1.275.1349.0    1448
1.275.941.0     1414
1.275.1362.0    1413
1.273.975.0     1400
Name: AvSigVersion, dtype: int64
```

```
df_virus2['AvSigVersion'] = df_virus2['AvSigVersion'].str.slice(stop=4)
```

```
df_virus2['AvSigVersion'].value_counts()
```

```
1.27    447343
1.26     34152
1.25      9678
1.23      5539
1.24      2372
1.22       896
1.20         7
1.21         5
0.0.         4
1.19         3
1.16         1
Name: AvSigVersion, dtype: int64
```

```
def setOthers(dataframe, column, num_values):
    top_categories = dataframe[column].value_counts().head(num_values)
    top_categories_list = top_categories.index.to_list()
    top_categories_list.append('Others')
    dataframe[column] = pd.Categorical(dataframe[column], categories=top_categories_list)
    return dataframe[column].fillna('Others')
```

```
df_virus2['AvSigVersion'] = setOthers(df_virus2, 'AvSigVersion', 2)
```

```
df_virus2['AvSigVersion'].value_counts()
```

```
1.27      447343
1.26       34152
Others     18505
Name: AvSigVersion, dtype: int64
```

Platform

```
df_virus2['Platform'].isnull().sum()
```

```
0
```

```
df_virus2['Platform'].value_counts()
```

```
windows10      483048
windows8        10825
windows7         5314
windows2016       813
Name: Platform, dtype: int64
```

```
df_virus2['Platform'] = df_virus2['Platform'].replace('windows8', 'Old platforms')
df_virus2['Platform'] = df_virus2['Platform'].replace('windows7', 'Old platforms')
df_virus2['Platform'] = df_virus2['Platform'].replace('windows2016', 'Old platforms')
```

```
df_virus2['Platform'].value_counts()
```

```
windows10        483048
Old platforms     16952
Name: Platform, dtype: int64
```

Processor

```
df_virus2['Processor'].isnull().sum()
```

```
0
```

```
df_virus2['Processor'].value_counts()
```

```
x64      454423
x86       45563
arm64        14
Name: Processor, dtype: int64
```

```
df_virus2['Processor'] = df_virus2['Processor'].replace('arm64', 'x64')
```

```
df_virus2['Processor'].value_counts()
```

```
x64    454437
x86     45563
Name: Processor, dtype: int64
```

OsVer

```
df_virus2['OsVer'].isnull().sum()
```

```
0
```

```
df_virus2['OsVer'].value_counts()
```

```
10.0.0.0      483830
6.3.0.0        10818
6.1.1.0         5281
6.1.0.0           33
10.0.3.0          12
10.0.1.0           7
6.3.3.0            2
10.0.0.1           2
6.3.1.0            2
10.0.32.72         2
10.0.32.0          1
10.0.80.0          1
10.0.5.0           1
10.0.2.0           1
```

```
6.3.5.0           1
10.0.4.0          1
10.0.8.0          1
10.0.0.112        1
6.3.32.72         1
6.3.7.0           1
10.0.7.0          1
Name: OsVer, dtype: int64
```

```
df_virus2['OsVer'] = df_virus2['OsVer'].str.slice(stop=4)
```

```
df_virus2['OsVer'].value_counts()
```

```
10.0    483861
6.3.     10825
6.1.      5314
Name: OsVer, dtype: int64
```

```
df_virus2['OsVer'] = df_virus2['OsVer'].replace('6.3.', '6')
df_virus2['OsVer'] = df_virus2['OsVer'].replace('6.1.', '6')
```

```
df_virus2['OsVer'].value_counts()
```

```
10.0    483861
6        16139
Name: OsVer, dtype: int64
```

OsPlatformSubRelease

```
df_virus2['OsPlatformSubRelease'].isnull().sum()
```

```
0
```

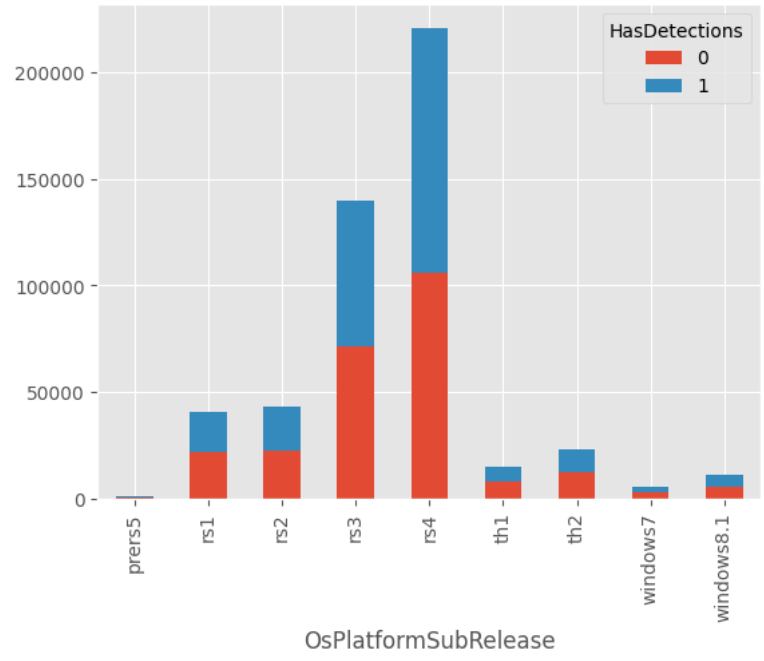
```
df_virus2['OsPlatformSubRelease'].value_counts()
```

```
rs4           220779
rs3           139901
rs2            43352
rs1            40717
th2            22955
th1            15014
windows8.1     10825
windows7        5314
prers5          1143
Name: OsPlatformSubRelease, dtype: int64
```

```
tabla_contingencia2 = pd.crosstab(df_virus2['OsPlatformSubRelease'], df_virus2['HasDetections'])
```

```
tabla_contingencia2.plot(kind='bar', stacked=True)
```

```
<Axes: xlabel='OsPlatformSubRelease'>
```



```
# All OsPlatformSubRelease types have a target average similar to that of the entire dataset (50%). We will leave the variable like this
```

OsBuildLab

```
df_virus2['OsBuildLab'].isnull().sum()
```

```
1
```

```
df_virus2['OsBuildLab'].value_counts()
```

```
17134.1.amd64fre.rs4_release.180410-1804                206436
16299.431.amd64fre.rs3_release_svc_escrow.180502-1908    69884
16299.15.amd64fre.rs3_release.170928-1534                53826
15063.0.amd64fre.rs2_release.170317-1834                 39908
17134.1.x86fre.rs4_release.180410-1804                   14334
                                                         ...
17604.1000.amd64fre.rs_prerelease.180209-1422                1
9600.17041.amd64fre.winblue_gdr.140305-1710                  1
9600.18146.x86fre.winblue_ltsb.151121-0600                   1
18214.1000.amd64fre.rs_prerelease.180803-1553                1
9600.18007.amd64fre.winblue_ltsb.150807-0612                 1
Name: OsBuildLab, Length: 453, dtype: int64
```

```
df_virus2.dropna(subset=['OsBuildLab'], inplace=True)
```

```
df_virus2['OsBuildLab'].isnull().sum()
```

```
0
```

```
df_virus2['OsBuildLab'].value_counts().head(20)
```

```
17134.1.amd64fre.rs4_release.180410-1804                206436
16299.431.amd64fre.rs3_release_svc_escrow.180502-1908    69884
16299.15.amd64fre.rs3_release.170928-1534                53826
15063.0.amd64fre.rs2_release.170317-1834                 39908
17134.1.x86fre.rs4_release.180410-1804                   14334
16299.15.x86fre.rs3_release.170928-1534                  13105
14393.2189.amd64fre.rs1_release.180329-1711              10724
10240.17443.amd64fre.th1.170602-2340                      9538
10586.1176.amd64fre.th2_release_sec.170913-1848           8385
15063.0.x86fre.rs2_release.170317-1834                    3444
14393.0.amd64fre.rs1_release.160715-1616                  3239
9600.19067.amd64fre.winblue_ltsb_escrow.180619-2033       3160
9600.19101.amd64fre.winblue_ltsb_escrow.180718-1800       3139
16299.637.amd64fre.rs3_release_svc.180808-1748            2439
14393.2189.x86fre.rs1_release.180329-1711                 2231
10586.1176.x86fre.th2_release_sec.170913-1848             2005
14393.693.amd64fre.rs1_release.161220-1747                2001
10240.17443.x86fre.th1.170602-2340                        1847
14393.2214.amd64fre.rs1_release_1.180402-1758             1702
10586.0.amd64fre.th2_release.151029-1700                  1491
Name: OsBuildLab, dtype: int64
```

```
df_virus2['OsBuildLab'] = df_virus2['OsBuildLab'].str.slice(stop=5)
```

```
df_virus2['OsBuildLab'].value_counts()
```

```
17134    220777
16299    139901
15063     43352
14393     40717
10586     22955
10240     15014
9600.     10825
7601.      5281
17692       155
17738       134
17744       129
17758        86
17746        62
17713        60
17754        56
17751        53
17741        51
17755        45
17735        37
17686        36
17733        34
17763        33
7600.        33
17760        31
```

```
17682        20
17677        17
18234        16
17672        15
18242        12
18237        12
17661         9
17666         8
17723         4
17634         4
17747         3
17618         3
17639         3
17730         3
17650         2
17655         2
17753         1
18214         1
17604         1
17711         1
17704         1
18219         1
17623         1
17749         1
14453         1
Name: OsBuildLab, dtype: int64
```

```
df_virus2['OsBuildLab'] = df_virus2['OsBuildLab'].str.slice(stop=2)
```

```
df_virus2['OsBuildLab'].value_counts()
```

```
17    221878
16    139901
15     43352
14     40718
10     37969
96     10825
76      5314
18        42
Name: OsBuildLab, dtype: int64
```

```
# In order, versions '18XX', '76XX' and '96XX' are the oldest. As they have few records, we will put them together and perform OHE once a
```

```
df_virus2['OsBuildLab'] = df_virus2['OsBuildLab'].replace('96', 'Older Build Lab')
df_virus2['OsBuildLab'] = df_virus2['OsBuildLab'].replace('76', 'Older Build Lab')
df_virus2['OsBuildLab'] = df_virus2['OsBuildLab'].replace('18', 'Older Build Lab')
```

```
df_virus2['OsBuildLab'].value_counts()
```

```
17                221878
16                139901
15                 43352
14                 40718
10                 37969
Older Build Lab    16181
Name: OsBuildLab, dtype: int64
```

['MachineIdentifier', 'ProductName', 'EngineVersion', 'AppVersion', 'AvSigVersion', 'Platform', 'Processor', 'OsVer', 'OsPlatformSubRelease', 'OsBuildLab', 'SkuEdition', 'PuaMode', 'SmartScreen', 'Census_MDC2FormFactor', 'Census_DeviceFamily', 'Census_ProcessorClass', 'Census_PrimaryDiskTypeName', 'Census_ChassisTypeName', 'Census_PowerPlatformRoleName', 'Census_InternalBatteryType', 'Census_OSVersion', 'Census_OSArchitecture', 'Census_OSBranch', 'Census_OSEdition', 'Census_OSSkuName', 'Census_OSInstallTypeName', 'Census_OSWUAutoUpdateOptionsName', 'Census_GenuineStateName', 'Census_ActivationChannel', 'Census_FlightRing']

```
def setOthers(dataframe, column, num_values):
    top_categories = dataframe[column].value_counts().head(num_values)
    top_categories_list = top_categories.index.to_list()
    top_categories_list.append('Others')
    dataframe[column] = pd.Categorical(dataframe[column], categories=top_categories_list)
    return dataframe[column].fillna('Others')
```

Census_OsVersion

```
df_virus2['Census_OSVersion'].isnull().sum()
```

```
0
```

```
df_virus2['Census_OSVersion'].value_counts()
```

```
10.0.17134.228     79975
10.0.17134.165     50511
10.0.16299.431     30519
10.0.17134.285     26289
10.0.17134.112     19501
                   ...
10.0.18214.1000        1
10.0.15019.1000        1
10.0.14328.1000        1
10.0.10586.601         1
10.0.15048.0           1
Name: Census_OSVersion, Length: 305, dtype: int64
```

```
df_virus2['Census_OSVersion'].value_counts().head(50)
```

```
10.0.17134.228      79975
10.0.17134.165      50511
10.0.16299.431      30519
10.0.17134.285      26289
10.0.17134.112      19501
10.0.16299.547      19179
10.0.16299.371      18195
10.0.17134.191      12989
10.0.14393.2189     12409
10.0.16299.611      12267
10.0.16299.125      12038
10.0.10240.17443    11476
10.0.10586.1176     10235
10.0.16299.492       9467
10.0.16299.309       8293
10.0.17134.286       7840
10.0.16299.15        6717
10.0.17134.254       6292
10.0.15063.1206      5700
10.0.17134.1         5678
10.0.15063.1266      5642
10.0.16299.192       5563
10.0.17134.167       4842
10.0.17134.137       4420
10.0.16299.248       4282
10.0.17134.48        3808
10.0.15063.0         3619
10.0.15063.1088      3535
10.0.14393.0         3311
10.0.17134.81        3044
10.0.14393.693       2801
10.0.15063.1155      2565
10.0.10586.164       2322
10.0.10586.318       2257
10.0.10586.0         2242
10.0.14393.2214      2183
10.0.15063.786       2139
10.0.16299.665       2118
10.0.15063.674       1961
10.0.15063.1324      1908
10.0.10586.494       1807
10.0.15063.850       1755
10.0.15063.726       1729
10.0.15063.483       1660
10.0.14393.447       1642
10.0.14393.1593      1543
10.0.14393.2007      1506
10.0.10586.420       1500
10.0.10240.16384     1499
10.0.15063.608       1421
Name: Census_OSVersion, dtype: int64
```

```
df_virus2['Census_OSVersion'].value_counts().tail(50)
```

```
10.0.14393.1613     2
10.0.10586.79       2
10.0.14986.1001     2
10.0.10586.842      2
10.0.10240.16389    2
10.0.10240.16387    2
10.0.14393.103      2
10.0.17004.1000     1
10.0.14295.1005     1
10.0.18219.1000     1
10.0.10586.1177     1
10.0.14926.1000     1
10.0.15063.448      1
10.0.14393.1737     1
10.0.17134.281      1
10.0.17623.1002     1
10.0.17749.1000     1
```

```
10.0.14421.191      1
10.0.14971.1000     1
10.0.17723.1000     1
10.0.14393.1670     1
10.0.16299.0        1
10.0.10240.16399    1
10.0.16193.1001     1
10.0.17618.1000     1
10.0.17046.1000     1
10.0.11082.1000     1
10.0.17604.1000     1
10.0.14393.1230     1
10.0.15042.0        1
10.0.14393.2156     1
10.0.17753.1        1
10.0.10240.17446    1
10.0.10240.17643    1
10.0.17074.1002     1
10.0.10586.456      1
6.3.9600.19069      1
10.0.14393.206      1
10.0.14946.1000     1
10.0.16251.1000     1
10.0.16299.491      1
10.0.16299.201      1
10.0.15063.1149     1
10.0.16299.1003     1
10.0.10240.16566    1
10.0.18214.1000     1
10.0.15019.1000     1
10.0.14328.1000     1
10.0.10586.601      1
10.0.15048.0        1
Name: Census_OSVersion, dtype: int64
```

```
# We will combine all records filtering by the first 7 digits
```

```
df_virus2['Census_OSVersion'] = df_virus2['Census_OSVersion'].str.slice(stop=7)
```

```
df_virus2['Census_OSVersion'] = df_virus2['Census_OSVersion'].replace('10.0.18', 'Other')
df_virus2['Census_OSVersion'] = df_virus2['Census_OSVersion'].replace('10.0.11', 'Other')
df_virus2['Census_OSVersion'] = df_virus2['Census_OSVersion'].replace('6.3.960', 'Other')
```

```
df_virus2['Census_OSVersion'].value_counts()
```

```
10.0.17    227106
10.0.16    136574
10.0.10     48299
10.0.15     44265
10.0.14     43710
Other          45
Name: Census_OSVersion, dtype: int64
```

Census_OSArchitecture

```
df_virus2['Census_OSArchitecture'].isnull().sum()
```

```
0
```

```
df_virus2['Census_OSArchitecture'].value_counts()
```

```
amd64    454434
x86       45551
arm64        14
Name: Census_OSArchitecture, dtype: int64
```

```
# We will put 'amd64' and 'arm64' together, since they have a similar naming and 'arm64' has very few values
```

```
df_virus2['Census_OSArchitecture'] = df_virus2['Census_OSArchitecture'].replace('arm64', 'amd64')
```

```
df_virus2['Census_OSArchitecture'].value_counts()
```

```
amd64    454448
x86       45551
Name: Census_OSArchitecture, dtype: int64
```

Census_OSBranch

```
df_virus2['Census_OSBranch'].isnull().sum()
```

```
0
```

```
df_virus2['Census_OSBranch'].value_counts()
```

```
rs4_release                 226000
rs3_release                  69216
rs3_release_svc_escrow       67027
rs2_release                  44264
rs1_release                  43705
th2_release                  18298
th2_release_sec              14895
th1_st1                      10822
th1                           4284
rs5_release                    839
rs3_release_svc_escrow_im      329
rs_prerelease                  184
rs_prerelease_flt              132
rs5_release_sigma                3
winblue_ltsb_escrow              1
Name: Census_OSBranch, dtype: int64
```

```
# We will combine rs3_release_svc_escrow_im with rs3_release_svc_escrow since they have a similar naming, and rs3_release_svc_escrow_im h
# We will put together th1_st1 and th1 since they have a similar naming, and both have relatively few records (< 5% of the total dataset)
# We will put together th2_release_sec and th2_release since they have a similar naming, and both have relatively few records (< 5% of th
```

```
df_virus2['Census_OSBranch'] = df_virus2['Census_OSBranch'].replace('rs3_release_svc_escrow_im', 'rs3_release_svc_escrow')
df_virus2['Census_OSBranch'] = df_virus2['Census_OSBranch'].replace('th1', 'th1_st1')
df_virus2['Census_OSBranch'] = df_virus2['Census_OSBranch'].replace('th2_release_sec', 'th2_release')
```

```
df_virus2['Census_OSBranch'].value_counts()
```

```
rs4_release               226000
rs3_release                69216
rs3_release_svc_escrow     67356
rs2_release                44264
rs1_release                43705
th2_release                33193
th1_st1                    15106
rs5_release                  839
rs_prerelease                184
rs_prerelease_flt            132
rs5_release_sigma              3
winblue_ltsb_escrow            1
Name: Census_OSBranch, dtype: int64
```

```
# We will group the records with few values as "Others", since they are not representative of the total dataset
```

```
df_virus2['Census_OSBranch'] = setOthers(df_virus2, 'Census_OSBranch', 7)
```

```
df_virus2['Census_OSBranch'].value_counts()
```

```
rs4_release               226000
rs3_release                69216
rs3_release_svc_escrow     67356
rs2_release                44264
rs1_release                43705
th2_release                33193
th1_st1                    15106
Others                      1159
Name: Census_OSBranch, dtype: int64
```

```
filtro_appversion_rs3_release = df_virus2['Census_OSBranch'] == 'rs3_release'
```

```
df_virus2[filtro_appversion_rs3_release]['HasDetections'].mean()
```

```
0.46486361534905224
```

```
filtro_appversion_rs3_release_svc_escrow = df_virus2['Census_OSBranch'] == 'rs3_release_svc_escrow'
```

```
df_virus2[filtro_appversion_rs3_release_svc_escrow]['HasDetections'].mean()
```

```
0.5152324959914484
```

```
# The target average of the "rs3_release" records is quite different from that of "rs3_release_svc_escrow", so we will keep them separate
# We will perform OHE once the review of the variables is finished
```

Census_OSEdition

```
df_virus2['Census_OSEdition'].isnull().sum()
```

```
0
```

```
df_virus2['Census_OSEdition'].value_counts()
```

```
Core                        194469
Professional                175807
CoreSingleLanguage          108696
CoreCountrySpecific           9275
ProfessionalEducation         3100
Education                     2310
Enterprise                    2055
ProfessionalN                 1624
EnterpriseS                   1108
ServerStandard                 584
Cloud                          336
CoreN                          251
ServerStandardEval             151
EducationN                      61
ServerDatacenterEval            47
EnterpriseSN                    47
ServerSolution                  34
EnterpriseN                     28
ProfessionalEducationN          12
ProfessionalWorkstation          2
CloudN                           1
ProfessionalWorkstationN         1
Name: Census_OSEdition, dtype: int64
```

```
df_virus2['Census_OSEdition'] = setOthers(df_virus2, 'Census_OSEdition', 3)
```

```
df_virus2['Census_OSEdition'].value_counts()
```

```
Core                  194469
Professional          175807
CoreSingleLanguage    108696
Others                 21027
Name: Census_OSEdition, dtype: int64
```

Census_OSSkuName

```
df_virus2['Census_OSSkuName'].isnull().sum()
```

```
0
```

```
df_virus2['Census_OSSkuName'].value_counts()
```

```
CORE                             194464
PROFESSIONAL                     178946
CORE_SINGLELANGUAGE              108674
CORE_COUNTRYSPECIFIC               9257
EDUCATION                          2313
ENTERPRISE                         2063
PROFESSIONAL_N                     1634
ENTERPRISE_S                       1107
STANDARD_SERVER                     584
CLOUD                               332
CORE_N                              251
STANDARD_EVALUATION_SERVER          151
EDUCATION_N                          61
ENTERPRISE_S_N                       48
DATACENTER_EVALUATION_SERVER         47
SB_SOLUTION_SERVER                   34
ENTERPRISE_N                         28
PRO_WORKSTATION                       2
CLOUDN                                1
PRO_WORKSTATION_N                     1
UNLICENSED                            1
Name: Census_OSSkuName, dtype: int64
```

```
df_virus2[['Census_OSSkuName', 'Census_OSEdition']].head(20)
```

	Census_OSSkuName	Census_OSEdition
0	PROFESSIONAL	Professional
1	PROFESSIONAL	Professional
2	CORE	Core
3	CORE	Core
4	PROFESSIONAL	Professional
5	CORE	Core
6	CORE	Core
7	CORE_COUNTRYSPECIFIC	Others
8	CORE	Core
9	PROFESSIONAL	Professional
10	CORE	Core
11	CORE	Core
12	CORE_SINGLELANGUAGE	CoreSingleLanguage
13	CORE_SINGLELANGUAGE	CoreSingleLanguage
14	PROFESSIONAL	Professional
15	PROFESSIONAL	Professional
16	PROFESSIONAL	Professional

```
# 'Census_OSSkuName' and 'Census_OSEdition' repeat information. So we will remove 'Census_OSSkuName'
```

18 CORE_SINGLELANGUAGE CoreSingleLanguage

```
df_virus2.drop('Census_OSSkuName', axis = 1, inplace = True)
```

Census_OSInstallTypeName

```
df_virus2['Census_OSInstallTypeName'].isnull().sum()
```

```
0
```

```
df_virus2['Census_OSInstallTypeName'].value_counts()
```

```
UUPUpgrade        146780
IBSClean           92403
Update             88891
Upgrade            70013
Other              46960
Reset              36510
Refresh            11540
Clean               3885
CleanPCRefresh      3017
Name: Census_OSInstallTypeName, dtype: int64
```

```
# We will keep the variable as it is, we can OHE itself
```

Census_OSWUAutoUpdateOptionsName

```
df_virus2['Census_OSWUAutoUpdateOptionsName'].isnull().sum()
```

```
0
```

```
df_virus2['Census_OSWUAutoUpdateOptionsName'].value_counts()
```

```
FullAuto                                222481
UNKNOWN                                 140961
Notify                                  113507
AutoInstallAndRebootAtMaintenanceTime    20731
Off                                       1506
DownloadNotify                             813
Name: Census_OSWUAutoUpdateOptionsName, dtype: int64
```

```
# We'll bundle 'Off' and 'DownloadNotify' as Others, since they have few records
```

```
df_virus2['Census_OSWUAutoUpdateOptionsName'] = df_virus2['Census_OSWUAutoUpdateOptionsName'].replace('Off', 'Others')
df_virus2['Census_OSWUAutoUpdateOptionsName'] = df_virus2['Census_OSWUAutoUpdateOptionsName'].replace('DownloadNotify', 'Others')
```

```
df_virus2['Census_OSWUAutoUpdateOptionsName'].value_counts()
```

```
FullAuto                                222481
UNKNOWN                                 140961
Notify                                  113507
AutoInstallAndRebootAtMaintenanceTime    20731
Others                                    2319
Name: Census_OSWUAutoUpdateOptionsName, dtype: int64
```

Census_GenuineStateName

```
df_virus2['Census_GenuineStateName'].isnull().sum()
```

```
0
```

```
df_virus2['Census_GenuineStateName'].value_counts()
```

```
IS_GENUINE         441401
INVALID_LICENSE     44990
OFFLINE             12834
UNKNOWN               774
Name: Census_GenuineStateName, dtype: int64
```

Census_ActivationChannel

```
df_virus2['Census_ActivationChannel'].isnull().sum()
```

```
0
```

```
df_virus2['Census_ActivationChannel'].value_counts()
```

```
Retail            264932
OEM:DM            191350
Volume:GVLK        25108
OEM:NONSLP         17943
Volume:MAK           468
Retail:TB:Eval       198
Name: Census_ActivationChannel, dtype: int64
```

```
# We will join 'Retail:TB:Eval' with 'Retail' , 'Volume:MAK' with 'Volume:GVLK'
```

```
df_virus2['Census_ActivationChannel'] = df_virus2['Census_ActivationChannel'].replace('Retail:TB:Eval', 'Retail')
df_virus2['Census_ActivationChannel'] = df_virus2['Census_ActivationChannel'].replace('Volume:MAK', 'Volume')
df_virus2['Census_ActivationChannel'] = df_virus2['Census_ActivationChannel'].replace('Volume:GVLK', 'Volume')
```

```
df_virus2['Census_ActivationChannel'].value_counts()
```

```
Retail        265130
OEM:DM        191350
Volume         25576
OEM:NONSLP     17943
Name: Census_ActivationChannel, dtype: int64
```

Census_FlightRing

```
df_virus2['Census_FlightRing'].isnull().sum()
```

```
0
```

```
df_virus2['Census_FlightRing'].value_counts()
```

```
Retail      468298
NOT_SET      16044
Unknown      13701
WIS            606
RP             583
WIF            549
Disabled       217
OSG              1
Name: Census_FlightRing, dtype: int64
```

```
# Everything that is not 'Retail' or 'NOT_SET', we will group it as Others
```

```
df_virus2['Census_FlightRing'] = setOthers(df_virus2, 'Census_FlightRing', 2)
```

```
df_virus2['Census_FlightRing'].value_counts()
```

```
Retail     468298
NOT_SET     16044
Others      15657
Name: Census_FlightRing, dtype: int64
```

Census_PrimaryDiskTypeName

```
df_virus2['Census_PrimaryDiskTypeName'].isnull().sum()
```

```
709
```

```
df_virus2['Census_PrimaryDiskTypeName'].value_counts()
```

```
HDD            325429
SSD            138155
UNKNOWN         20082
Unspecified     15624
Name: Census_PrimaryDiskTypeName, dtype: int64
```

```
# We will assign the nulls and the values of 'UNKNOWN' & 'Unspecified' to an 'Others' column
```

```
df_virus2['Census_PrimaryDiskTypeName'] = setOthers(df_virus2, 'Census_PrimaryDiskTypeName', 2)
```

```
df_virus2['Census_PrimaryDiskTypeName'].isnull().sum()
```

```
0
```

```
df_virus2['Census_PrimaryDiskTypeName'].value_counts()
```

```
HDD       325429
SSD       138155
Others     36415
Name: Census_PrimaryDiskTypeName, dtype: int64
```

Census_ChassisTypeName

```
df_virus2['Census_ChassisTypeName'].isnull().sum()
```

```
37
```

```
df_virus2['Census_ChassisTypeName'].value_counts()
```

```
Notebook               294232
Desktop                104978
Laptop                  38261
Portable                20181
AllinOne                11407
MiniTower                4849
Convertible              4685
Other                    4215
UNKNOWN                  3695
Detachable               2930
LowProfileDesktop        2878
HandHeld                 2652
SpaceSaving              1689
Tablet                    730
Tower                     692
Unknown                   575
MainServerChassis         512
MiniPC                    261
LunchBox                  224
RackMountChassis          189
SubNotebook                47
BusExpansionChassis        38
30                         11
StickPC                     7
0                           5
MultisystemChassis          3
35                          3
PizzaBox                    3
Blade                       3
31                          2
SubChassis                  2
32                          1
ExpansionChassis            1
25                          1
Name: Census_ChassisTypeName, dtype: int64
```

```
# We will join 'LowProfileDesktop' to 'Desktop'

df_virus2['Census_ChassisTypeName'] = df_virus2['Census_ChassisTypeName'].replace('LowProfileDesktop', 'Desktop')

# The rest of the columns, along with the null ones, will be grouped in a column called 'Others'

df_virus2['Census_ChassisTypeName'].value_counts()
```

```
Notebook               294232
Desktop                107856
Laptop                  38261
Portable                20181
AllinOne                11407
MiniTower                4849
Convertible              4685
Other                    4215
UNKNOWN                  3695
Detachable               2930
HandHeld                 2652
SpaceSaving              1689
Tablet                    730
Tower                     692
Unknown                   575
MainServerChassis         512
MiniPC                    261
LunchBox                  224
RackMountChassis          189
SubNotebook                47
BusExpansionChassis        38
30                         11
StickPC                     7
0                           5
MultisystemChassis          3
35                          3
PizzaBox                    3
Blade                       3
31                          2
SubChassis                  2
32                          1
ExpansionChassis            1
25                          1
Name: Census_ChassisTypeName, dtype: int64
```

```
df_virus2[['Census_ChassisTypeName', 'Census_MDC2FormFactor']].head(20)
```

	Census_ChassisTypeName	Census_MDC2FormFactor
0	AllinOne	Desktop
1	Notebook	Notebook
2	Notebook	Convertible
3	Notebook	Notebook
4	Portable	Notebook
5	Portable	Notebook
6	Notebook	Notebook
7	Laptop	Notebook
8	Laptop	Notebook
9	Notebook	Notebook
10	AllinOne	AllInOne
11	Notebook	Notebook
12	AllinOne	AllInOne
13	Notebook	Notebook
14	Notebook	Notebook
15	Notebook	Notebook
16	Notebook	Desktop
17	Convertible	Convertible
18	Notebook	Notebook
19	Desktop	Desktop

```
df_virus2['Census_ChassisTypeName'] = setOthers(df_virus2, 'Census_ChassisTypeName', 4)
```

```
df_virus2["Census_ChassisTypeName"].value_counts()
```

```
Notebook    294232
Desktop     107856
Others       39469
Laptop       38261
Portable     20181
Name: Census_ChassisTypeName, dtype: int64
```

```
df_virus3 = df_virus2.copy()
```

SkuEdition

```
df_virus3['SkuEdition'].value_counts()
```

```
Home               308567
Pro                181041
Invalid              4423
Education            2321
Enterprise           1999
Enterprise LTSB      1141
Cloud                 309
Server                198
Name: SkuEdition, dtype: int64
```

```
df_virus3['SkuEdition'] = df_virus3['SkuEdition'].replace('Enterprise LTSB', 'Enterprise') # Juntamos estas dos variables en una al tener
```

```
df_virus3['SkuEdition'].value_counts()
```

```
Home          308567
Pro           181041
Invalid         4423
Enterprise      3140
Education       2321
Cloud            309
Server           198
Name: SkuEdition, dtype: int64
```

```
df_virus3['SkuEdition'] = setOthers(df_virus3, 'SkuEdition', 2) #vamos a juntar las variables que no son Home y Pro por ser pocos represe
```

PuaMode

```
del(df_virus3["PuaMode"]) #Eliminamos la columna por no tener suficientes valores
```

SmartScreen

```
df_virus3['SmartScreen'].value_counts()
```

```
RequireAdmin    241593
ExistsNotSet     58497
Off              10388
Warn              7530
Prompt            1950
Block             1274
off                 75
On                  53
&#x02;              20
&#x01;              14
on                   8
requireadmin         1
Name: SmartScreen, dtype: int64
```

```
df_virus3['SmartScreen'] = df_virus3['SmartScreen'].replace('requireadmin', 'RequireAdmin') #Juntamos variables porque tienen el mismo nc
```

```
df_virus3['SmartScreen'].fillna('No info', inplace = True) #remplazamos los nulos a una variable sin información
```

```
df_virus3['SmartScreen'] = setOthers(df_virus3, 'SmartScreen', 2)
```

Census_MDC2FormFactor

```
df_virus3['Census_MDC2FormFactor'].value_counts()
```

```
Notebook        320948
Desktop         109526
Convertible      22369
Detachable       16802
AllInOne         16372
PCOther           7800
LargeTablet       3645
SmallTablet       1797
SmallServer        496
MediumServer       192
LargeServer         50
ServerOther          2
Name: Census_MDC2FormFactor, dtype: int64
```

```
df_virus3['Census_MDC2FormFactor'] = setOthers(df_virus3, 'Census_MDC2FormFactor', 2) #Juntamos las variables en Notebook, Desktop y otro
```

Census_DeviceFamily

```
df_virus3['Census_DeviceFamily'].value_counts()
```

```
Windows.Desktop    499182
Windows.Server        816
Windows                 1
Name: Census_DeviceFamily, dtype: int64
```

```
del(df_virus3["Census_DeviceFamily"])
```

Census_ProcessorClass

```
df_virus3['Census_ProcessorClass'].value_counts()
```

```
mid     1196
low      546
high     340
Name: Census_ProcessorClass, dtype: int64
```

```
del(df_virus3["Census_ProcessorClass"]) #Eliminamos porque tiene muchos nulos
```

Census_PowerPlatformRoleName

```
df_virus3['Census_PowerPlatformRoleName'].value_counts()
```

```
Mobile               346378
Desktop              116053
Slate                 27475
Workstation            6235
SOHOServer             2062
UNKNOWN                1172
EnterpriseServer        406
AppliancePC             212
PerformanceServer         4
Name: Census_PowerPlatformRoleName, dtype: int64
```

```
df_virus3['Census_PowerPlatformRoleName'].fillna('UNKNOWN', inplace = True) #Juntamos los nulos con la columna UNKNOWN
```

```
df_virus3['Census_PowerPlatformRoleName'] = setOthers(df_virus3, 'Census_PowerPlatformRoleName', 2) #Dejamos solo Mobile y Desktop como r
```

Census_InternalBatteryType

```
df_virus3['Census_InternalBatteryType'].value_counts()
```

```
lion    113500
li-i     13855
#        10175
lip       3326
liio      1814
li p       466
li         356
nimh       272
real       162
bq20       143
pbac       130
vbox        89
lgi0        29
unkn        19
lipo        13
```

```
lhp0        12
ithi         7
4cel         6
ram          5
batt         3
bad          3
lipp         3
a132         2
virt         2
li-l         1
lgl0         1
icp3         1
3ion         1
Name: Census_InternalBatteryType, dtype: int64
```

```
del(df_virus3["Census_InternalBatteryType"]) #Eliminating because it has a lot of nulls
```

Haz doble clic (o pulsa Intro) para editar

We analyze the boolean variables

1. First a summary of the columns that have nulls

```
df_virus2[columnas_booleanas].isnull().sum()

#First check to see if there are many nulls or not in the boolean columns

#IsBeta has 1 "1" -> we remove it since there is only 1 row with a different value from the rest
#AutoSampleOptIn has 14 "1" -> we remove it for the same reason above + it has values of "1" that are outside the 0-1 range of the TARGET
#Census_IsFlightingInternal has 2 "1" & +415k nulls -> we remove because it has one row with positive detection, and the other row with r
#Census_ThresholdOptIn has 42 "1" & +315k nulls -> we eliminate because the 47 have both 0 and 1 in the target, so it is not relevant to


### Less than 300 rows with "1"
##SMode
##Census_IsPortableOperatingSystem

### Has all its "1's" that give the same target detection result
##Census_IsFlightsDisabled all 7 "1"s give the target 0
```

```
IsBeta                                      0
IsSxsPassiveMode                            0
HasTpm                                      0
IsProtected                              1926
AutoSampleOptIn                             0
SMode                                   29848
Firewall                                 5162
Census_HasOpticalDiskDrive                  0
Census_IsPortableOperatingSystem            0
Census_IsFlightingInternal                415225
Census_IsFlightsDisabled                   8933
Census_ThresholdOptIn                     318104
Census_IsSecureBootEnabled                  0
Census_IsVirtualDevice                    901
Census_IsTouchEnabled                       0
Census_IsPenCapable                         0
Census_IsAlwaysOnAlwaysConnectedCapable     4040
Wdft_IsGamer                              16950
dtype: int64
```

2. We´ll generate a copy to work only with booleans

```
df_virus4 = df_virus3.copy()
```

3.1 IsBeta

```
df_virus4["IsBeta"].value_counts()
```

```
0    499998
1         1
Name: IsBeta, dtype: int64
```

```
del(df_virus4["IsBeta"])
```

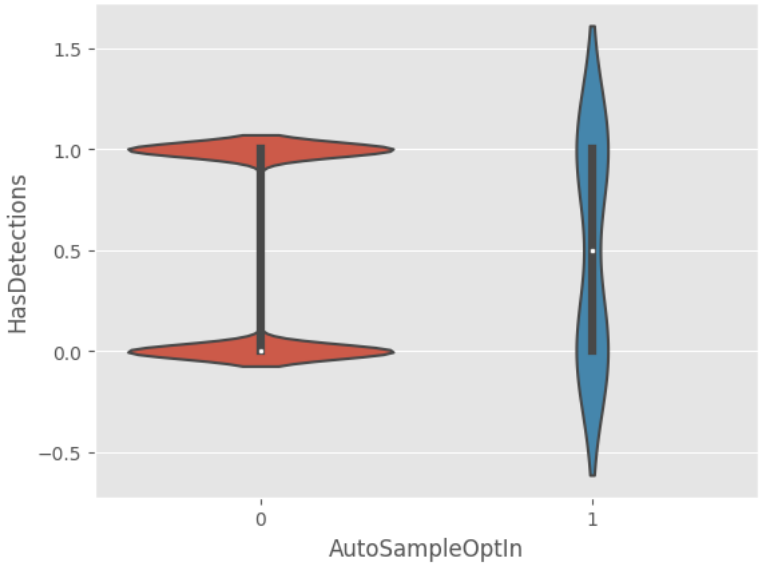
3.2 AutoSampleOptIn

```
df_virus4["AutoSampleOptIn"].value_counts()
```

```
0    499985
1        14
Name: AutoSampleOptIn, dtype: int64
```

```
sns.violinplot(x="AutoSampleOptIn", y=TARGET, data=df_virus4)
```

```
<Axes: xlabel='AutoSampleOptIn', ylabel='HasDetections'>
```




```
del(df_virus4["AutoSampleOptIn"])
```

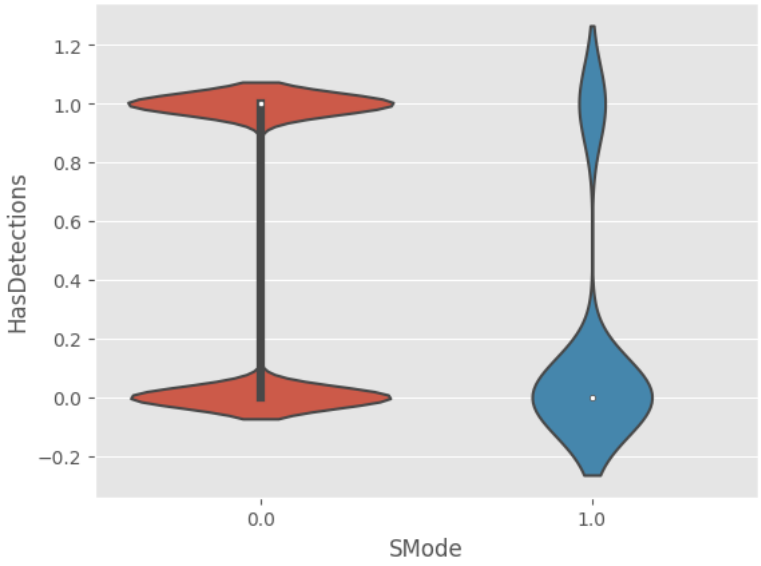
3.3 SMode

```
df_virus4["SMode"].value_counts()
```

```
0.0    469945
1.0       206
Name: SMode, dtype: int64
```

```
sns.violinplot(x="SMode", y=TARGET, data=df_virus4)
```

```
<Axes: xlabel='SMode', ylabel='HasDetections'>
```

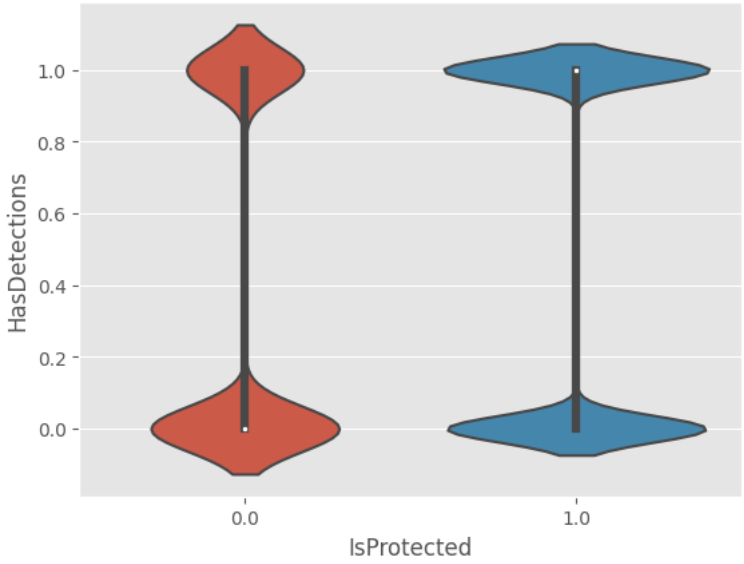



```
df_virus4["SMode"].fillna("3", inplace=True)  #Reemplazamos los nulos por valor "3" para diferenciarlos luego
```

3.4 IsProtected

```
sns.violinplot(x="IsProtected", y=TARGET, data=df_virus4)
```

```
<Axes: xlabel='IsProtected', ylabel='HasDetections'>
```




```
df_virus4["IsProtected"].fillna("3", inplace=True)  #Reemplazamos los nulos por valor "3" para diferenciarlos luego
```

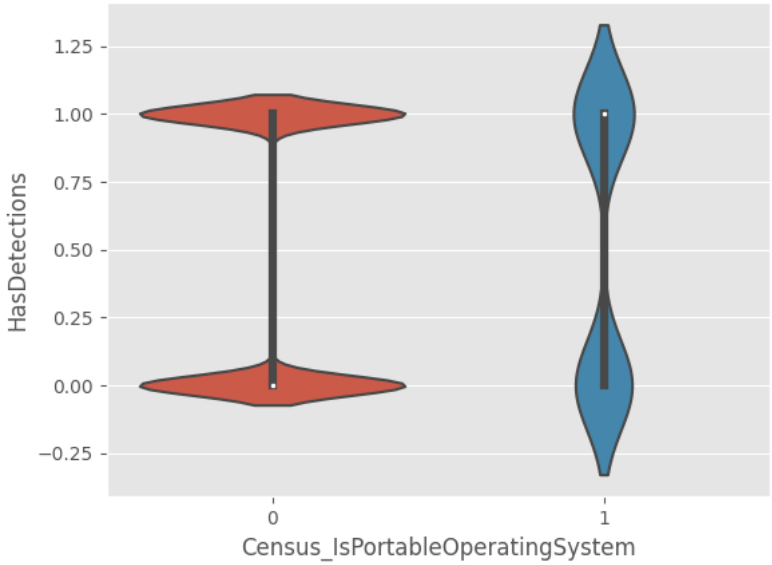
3.5 Census_IsPortableOperatingSystem

```
df_virus4["Census_IsPortableOperatingSystem"].value_counts()
```

```
0    499738
1       261
Name: Census_IsPortableOperatingSystem, dtype: int64
```

```
sns.violinplot(x="Census_IsPortableOperatingSystem", y=TARGET, data=df_virus4)
```

```
<Axes: xlabel='Census_IsPortableOperatingSystem', ylabel='HasDetections'>
```



3.6 Firewall

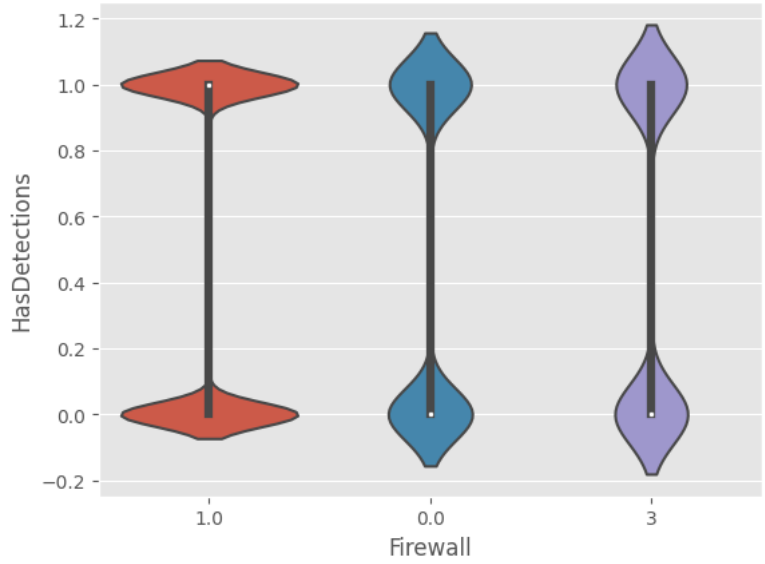
```
df_virus4["Firewall"].fillna("3", inplace=True)  #We replace the nulls with the value "3" to differentiate them later
```

```
df_virus4["Firewall"].value_counts()
```

```
1.0    484071
0.0     10766
3        5162
Name: Firewall, dtype: int64
```

```
sns.violinplot(x="Firewall", y=TARGET, data=df_virus4)
#We could group the 3 with the 0
```

```
<Axes: xlabel='Firewall', ylabel='HasDetections'>
```



3.7 Census_IsFlightingInternal

```
df_virus4["Census_IsFlightingInternal"].value_counts()
```

```
0.0    84772
1.0        2
Name: Census_IsFlightingInternal, dtype: int64
```

```
df_virus4[df_virus4["Census_IsFlightingInternal"] == 1]
```

	Unnamed: 0	ProductName	EngineVersion	AppVersion	AvSigVersion	RtpStateBitfield	IsSxsPassiveMode	DefaultBrowsersIdentifi
248891	2766127	win8defender	1.1.15	4.18	1.27	7.0	0	Na
326169	7832694	win8defender	1.1.15	Others	1.27	7.0	0	Na

2 rows × 76 columns

```
del(df_virus4["Census_IsFlightingInternal"])
```

3.8 Census_IsFlightsDisabled

```
df_virus4["Census_IsFlightsDisabled"].value_counts()
```

```
0.0    491059
1.0         7
Name: Census_IsFlightsDisabled, dtype: int64
```

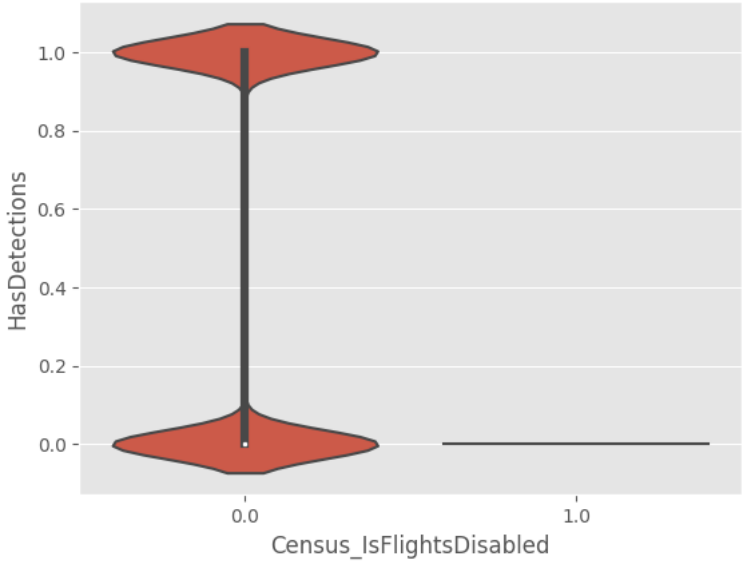
```
df_virus4[df_virus4["Census_IsFlightsDisabled"] == 1]
```

	Unnamed: 0	ProductName	EngineVersion	AppVersion	AvSigVersion	RtpStateBitfield	IsSxsPassiveMode	DefaultBrowsersIdentifi
21062	4303754	win8defender	1.1.15	4.18	1.27	7.0	0	Na
25521	597753	win8defender	1.1.15	Others	1.27	7.0	0	Na
177475	6745478	win8defender	1.1.15	4.18	1.27	7.0	0	Na
209872	2638051	win8defender	1.1.15	4.18	1.27	7.0	0	Na
284594	4060493	win8defender	1.1.15	Others	1.27	7.0	0	Na
287335	6767930	win8defender	1.1.15	4.18	1.27	7.0	0	Na
358641	5752548	win8defender	1.1.15	4.18	1.27	7.0	0	Na

7 rows × 75 columns

```
sns.violinplot(x="Census_IsFlightsDisabled", y=TARGET, data=df_virus4)
```

```
<Axes: xlabel='Census_IsFlightsDisabled', ylabel='HasDetections'>
```




```
df_virus4["Census_IsFlightsDisabled"].fillna("3", inplace=True)
#We replace the nulls with the value "3" to differentiate them later

#We could mix the 3 with the 0
```

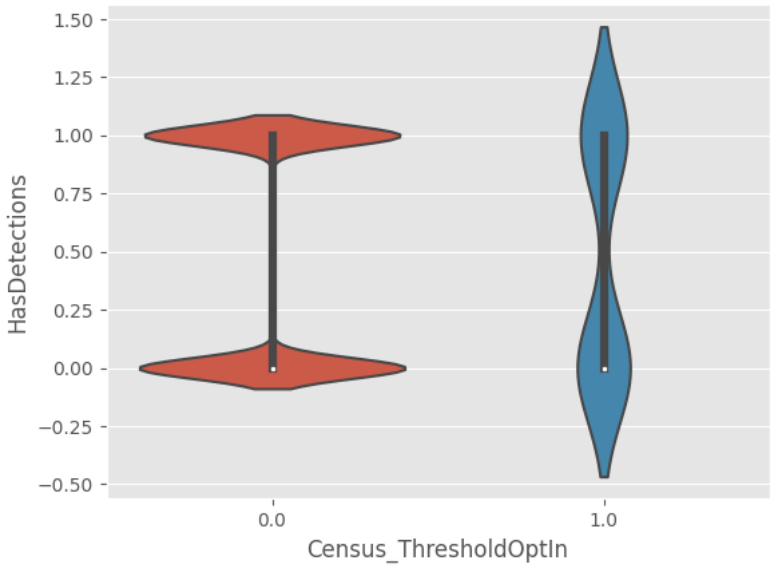
3.9 Census_ThresholdOptIn

```
df_virus4["Census_ThresholdOptIn"].value_counts()
```

```
0.0    181848
1.0        47
Name: Census_ThresholdOptIn, dtype: int64
```

```
sns.violinplot(x="Census_ThresholdOptIn", y=TARGET, data=df_virus4)
```

```
<Axes: xlabel='Census_ThresholdOptIn', ylabel='HasDetections'>
```




```
df_virus4[df_virus4["Census_ThresholdOptIn"] == 1]
```

	Unnamed: 0	ProductName	EngineVersion	AppVersion	AvSigVersion	RtpStateBitfield	IsSxsPassiveMode	DefaultBrowsersIdenti
15828	8365567	win8defender	1.1.15	4.18	1.27	7.0	0	
20165	7553142	win8defender	1.1.15	4.18	1.27	7.0	0	
32859	8035920	win8defender	1.1.15	4.18	1.27	7.0	0	
39905	1189053	win8defender	1.1.15	4.18	1.27	7.0	0	
44742	239461	win8defender	1.1.15	4.18	1.27	7.0	0	
67033	4640662	win8defender	1.1.15	4.18	1.27	7.0	0	
67740	1517632	win8defender	1.1.15	Others	1.27	7.0	0	
148654	5455573	win8defender	1.1.15	4.18	1.27	7.0	0	
183699	1660743	win8defender	1.1.15	4.18	1.27	7.0	0	
188160	912336	win8defender	1.1.15	4.18	1.27	7.0	0	
190636	6516014	win8defender	1.1.15	4.14	1.27	7.0	0	
191783	8781600	win8defender	1.1.15	4.18	1.27	7.0	0	
194412	4551988	win8defender	1.1.15	4.18	1.27	7.0	0	
198238	668643	win8defender	1.1.15	4.18	1.27	7.0	0	
224720	4525277	win8defender	1.1.15	4.18	1.27	7.0	0	
234413	8048900	win8defender	1.1.15	Others	1.27	7.0	0	
247768	2854614	win8defender	1.1.15	Others	1.27	7.0	0	
264460	4268480	win8defender	1.1.15	Others	1.27	7.0	0	
268261	7847731	win8defender	1.1.15	4.18	1.27	7.0	0	
278897	5943084	win8defender	1.1.14	4.14	1.26	7.0	0	
285004	3369662	win8defender	1.1.15	4.18	1.27	7.0	0	
289844	7861139	win8defender	1.1.15	4.18	1.27	7.0	0	
322467	8804978	win8defender	1.1.15	4.18	1.27	0.0	1	
327816	8548899	win8defender	1.1.15	4.18	1.27	7.0	0	
341772	3943321	win8defender	1.1.15	4.18	1.27	7.0	0	
343535	2890141	win8defender	1.1.15	4.18	1.27	7.0	0	
354110	2452286	win8defender	1.1.14	Others	1.26	7.0	0	
361756	4978236	win8defender	1.1.15	4.18	1.27	7.0	0	
363884	4814399	win8defender	1.1.14	Others	1.26	7.0	0	
365149	7471135	win8defender	1.1.15	4.18	1.27	7.0	0	
375226	4513257	win8defender	1.1.15	4.18	1.27	7.0	0	
377158	5072821	win8defender	1.1.15	Others	1.27	7.0	0	
386431	4335834	win8defender	1.1.15	4.18	1.27	7.0	0	
386910	4529879	win8defender	1.1.14	4.16	1.26	7.0	0	
396309	545718	win8defender	1.1.15	4.18	1.27	7.0	0	
397514	480386	win8defender	1.1.15	4.18	1.27	0.0	0	
400940	7596111	win8defender	1.1.15	4.18	1.27	7.0	0	
403256	8119995	win8defender	1.1.15	4.18	1.27	7.0	0	
411671	2154679	win8defender	1.1.15	4.18	1.27	7.0	0	
422829	281557	win8defender	1.1.15	4.18	1.27	7.0	0	
427764	6739576	win8defender	1.1.15	4.18	1.27	7.0	0	

```
del(df_virus4["Census_ThresholdOptIn"])
```

453978	7097016	win8defender	1.1.15	4.18	1.27	7.0	0	

4.0 Census_IsPenCapable

```
df_virus4["Census_IsPenCapable"].value_counts()
```

```
0    481127
1     18872
Name: Census_IsPenCapable, dtype: int64
```

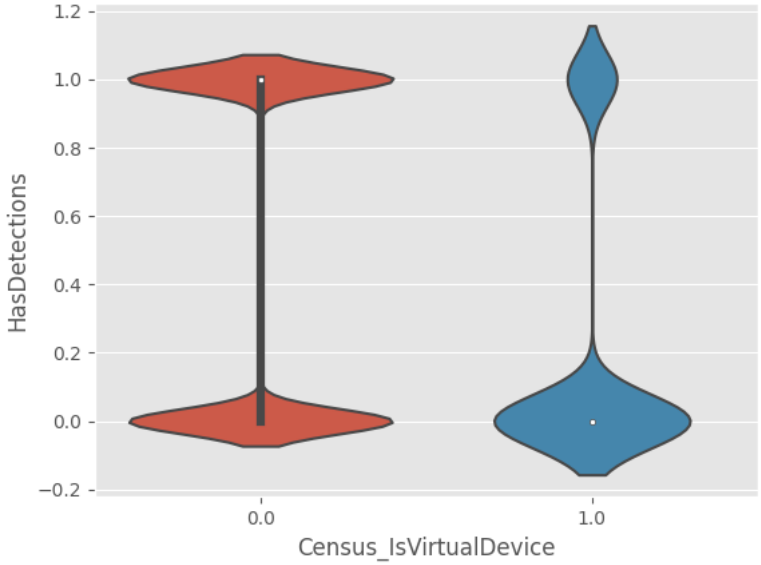
4.1 Census_IsVirtualDevice

```
df_virus4["Census_IsVirtualDevice"].value_counts()
```

```
0.0    495571
1.0      3527
Name: Census_IsVirtualDevice, dtype: int64
```

```
sns.violinplot(x="Census_IsVirtualDevice", y=TARGET, data=df_virus4)
```

```
<Axes: xlabel='Census_IsVirtualDevice', ylabel='HasDetections'>
```




```
df_virus4["Census_IsVirtualDevice"].fillna("3", inplace=True)  #Reemplazamos los nulos por valor "3" para diferenciarlos luego
```

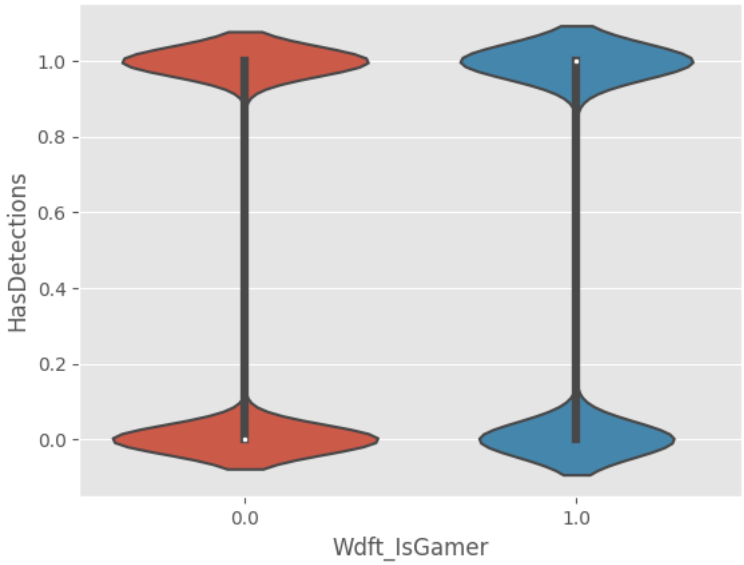
4.2 Wdft_IsGamer

```
df_virus4["Wdft_IsGamer"].value_counts()
```

```
0.0    345835
1.0    137214
Name: Wdft_IsGamer, dtype: int64
```

```
sns.violinplot(x="Wdft_IsGamer", y=TARGET, data=df_virus3)
```

```
<Axes: xlabel='Wdft_IsGamer', ylabel='HasDetections'>
```

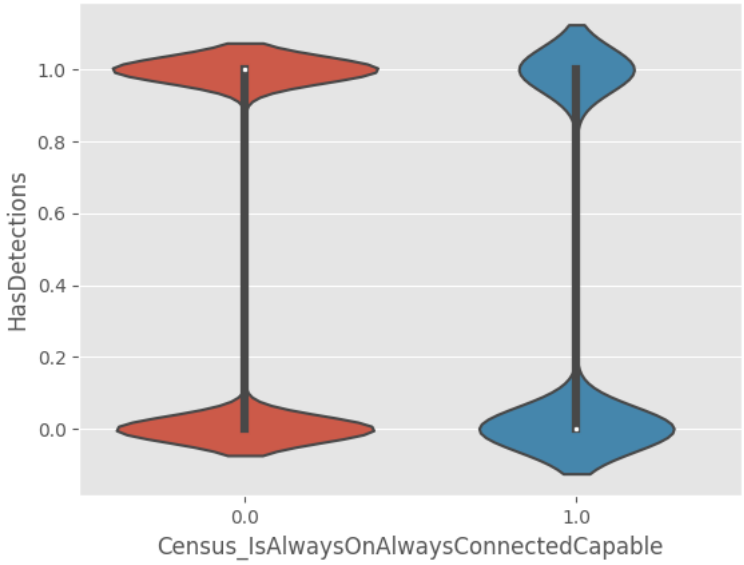



```
df_virus4["Wdft_IsGamer"].fillna("3", inplace=True)  #Reemplazamos los nulos por valor "3" para diferenciarlos luego
```

4.3 Census_IsAlwaysOnAlwaysConnectedCapable

```
sns.violinplot(x="Census_IsAlwaysOnAlwaysConnectedCapable", y=TARGET, data=df_virus4)
```

```
<Axes: xlabel='Census_IsAlwaysOnAlwaysConnectedCapable', ylabel='HasDetections'>
```




```
df_virus4["Census_IsAlwaysOnAlwaysConnectedCapable"].fillna("3", inplace=True)
#We replace the nulls with the value "3" to differentiate them later
```

Analyze the numeric variables

```
df_virus5 = df_virus4.copy()
```

```
df_virus5[columnas_numericas].isnull().sum()

#We first analyze those that do not have nulls to see if there are outliers
```

```
Unnamed: 0                                            0
RtpStateBitfield                                   1832
DefaultBrowsersIdentifier                        475938
AVProductStatesIdentifier                          1938
AVProductsInstalled                                1938
AVProductsEnabled                                  1938
CountryIdentifier                                     0
CityIdentifier                                    18239
OrganizationIdentifier                           154563
GeoNameIdentifier                                    16
LocaleEnglishNameIdentifier                           0
OsBuild                                               0
OsSuite                                               0
IeVerIdentifier                                    3209
UacLuaenable                                        623
Census_OEMNameIdentifier                           5381
Census_OEMModelIdentifier                          5764
Census_ProcessorCoreCount                          2347
Census_ProcessorManufacturerIdentifier             2347
Census_ProcessorModelIdentifier                    2349
Census_PrimaryDiskTotalCapacity                    2976
Census_SystemVolumeTotalCapacity                   2976
Census_TotalPhysicalRAM                            4556
Census_InternalPrimaryDiagonalDisplaySizeInInches  2654
Census_InternalPrimaryDisplayResolutionHorizontal  2650
Census_InternalPrimaryDisplayResolutionVertical    2650
Census_InternalBatteryNumberOfCharges             15038
Census_OSBuildNumber                                  0
Census_OSBuildRevision                                0
Census_OSInstallLanguageIdentifier                 3332
Census_OSUILocaleIdentifier                           0
Census_FirmwareManufacturerIdentifier             10349
Census_FirmwareVersionIdentifier                   9061
Census_IsWIMBootEnabled                          317666
Wdft_RegionIdentifier                             16950
dtype: int64
```

0. We identified several columns that are categorical and not numeric, so we will change their type

```
columnas_cambio_acateg = ["DefaultBrowsersIdentifier", "CountryIdentifier", "LocaleEnglishNameIdentifier", "OsBuild", "AVProductStatesIde
                          "CityIdentifier", "OrganizationIdentifier", "GeoNameIdentifier", "OsSuite", "IeVerIdentifier", "UacLuaenable",
                          "Census_OEMModelIdentifier", "Census_ProcessorCoreCount", "Census_ProcessorManufacturerIdentifier", "Census_Pro
```

```
                            "Census_SystemVolumeTotalCapacity", "Census_TotalPhysicalRAM", "Census_InternalPrimaryDiagonalDisplaySizeInInch
                            "Census_InternalPrimaryDisplayResolutionVertical", "Census_InternalBatteryNumberOfCharges" , "Census_OSBuildNum
                            "Census_OSInstallLanguageIdentifier", "Census_OSUILocaleIdentifier","Census_FirmwareManufacturerIdentifier","Ce
```

```
for column in columnas_cambio_acateg:
  df_virus5[column] = df_virus5[column].astype('object')
```

```
df_virus5["OsSuite"].dtype
```

```
dtype('O')
```

0. We group the variables that have a lot of data that is not significant and analyze them to see if we group or eliminate

```
df_virus5["DefaultBrowsersIdentifier"].value_counts()
```

```
239.0     2589
3195.0    2423
1632.0    1601
3176.0    1313
146.0     1231
          ...
2285.0       1
2100.0       1
2459.0       1
2196.0       1
1427.0       1
Name: DefaultBrowsersIdentifier, Length: 554, dtype: int64
```

```
del(df_virus5["DefaultBrowsersIdentifier"]) #We eliminate because it only has 24,000 data and the rest are null
```

```
df_virus5["CountryIdentifier"].value_counts()
```

```
43     22339
29     19273
141    18419
93     16205
171    15743
       ...
212       13
37        12
165       11
217       10
193        9
Name: CountryIdentifier, Length: 222, dtype: int64
```

```
del(df_virus5["CountryIdentifier"]) #We eliminate because the ID does not give us a reference to the country
```

```
df_virus5["LocaleEnglishNameIdentifier"].value_counts()
```

```
75     117818
182     25187
42      23007
74      22894
88      20988
        ...
152         1
155         1
165         1
190         1
127         1
Name: LocaleEnglishNameIdentifier, Length: 233, dtype: int64
```

```
del(df_virus5["LocaleEnglishNameIdentifier"]) #We eliminate because the ID does not give us a reference to the country
```

```
df_virus5["OsBuild"].value_counts()
```

```
17134    220778
16299    139901
15063     43352
14393     40717
10586     22955
10240     15014
9600      10825
7601       5281
17692       155
17738       134
17744       129
17758        86
17746        62
```

```
17713        60
17754        56
17751        53
17741        51
17755        45
17735        37
17686        36
17733        34
17763        33
7600         33
17760        31
17682        20
17677        17
18234        16
17672        15
18242        12
18237        12
17661         9
17666         8
17723         4
17634         4
17639         3
17730         3
17618         3
17747         3
17655         2
17650         2
17753         1
17604         1
18214         1
17711         1
17704         1
18219         1
17623         1
17749         1
Name: OsBuild, dtype: int64
```

```
df_virus5['OsBuild'] = df_virus5['OsBuild'].astype(str).str.slice(stop=2)
```

```
df_virus5["AVProductStatesIdentifier"].value_counts()
```

```
53447.0    326462
7945.0      26642
47238.0     18436
62773.0     14931
46413.0      6343
             ...
2451.0          1
19604.0         1
3193.0          1
26492.0         1
18324.0         1
Name: AVProductStatesIdentifier, Length: 5516, dtype: int64
```

```
df_virus5['AVProductStatesIdentifier'] = setOthers(df_virus5, 'AVProductStatesIdentifier', 7)
```

```
df_virus5["AVProductsInstalled"].value_counts()
```

```
1.0    348045
2.0    137856
3.0     11617
4.0       514
5.0        29
Name: AVProductsInstalled, dtype: int64
```

```
df_virus5['AVProductsInstalled'] = setOthers(df_virus5, 'AVProductsInstalled', 2)
```

```
df_virus5["AVProductsInstalled"].value_counts()
```

```
1.0       348045
2.0       137856
Others     14098
Name: AVProductsInstalled, dtype: int64
```

```
df_virus5["AVProductsEnabled"].value_counts()
```

```
1.0    485178
2.0     11074
0.0      1467
3.0       316
4.0        26
Name: AVProductsEnabled, dtype: int64
```

```
df_virus5['AVProductsEnabled'] = setOthers(df_virus5, 'AVProductsEnabled', 1)
```

```
df_virus5["AVProductsEnabled"].value_counts()
```

```
1.0       485178
Others     14821
Name: AVProductsEnabled, dtype: int64
```

```
df_virus5["CityIdentifier"].value_counts()
```

```
130775.0    5271
16668.0     4747
82373.0     4649
10222.0     4011
61668.0     3724
            ...
151632.0       1
41727.0        1
83230.0        1
131294.0       1
148194.0       1
Name: CityIdentifier, Length: 37157, dtype: int64
```

```
del(df_virus5["CityIdentifier"]) #We delete because the ID does not give us reference to the city
```

```
df_virus5["OrganizationIdentifier"].value_counts()
```

```
27.0    234884
18.0     98842
48.0      3658
50.0      2538
37.0      1048
11.0      1037
49.0       747
46.0       641
14.0       255
32.0       230
36.0       207
52.0       174
33.0       171
2.0        153
5.0        110
40.0       100
28.0        98
4.0         79
10.0        69
51.0        53
1.0         39
20.0        38
8.0         37
6.0         25
31.0        23
47.0        22
22.0        21
39.0        21
3.0         20
21.0        19
16.0        16
19.0        10
29.0         8
42.0         8
7.0          7
44.0         6
26.0         6
43.0         4
45.0         4
41.0         2
35.0         2
23.0         2
25.0         1
17.0         1
Name: OrganizationIdentifier, dtype: int64
```

```
del(df_virus5["OrganizationIdentifier"])
#We eliminate because the ID does not give us a reference to the organization
```

```
df_virus5["GeoNameIdentifier"].value_counts()
```

```
277.0    86080
211.0    23593
53.0     22977
89.0     20174
240.0    19489
         ...
```

```
280.0        1
231.0        1
113.0        1
221.0        1
132.0        1
Name: GeoNameIdentifier, Length: 268, dtype: int64
```

```
del(df_virus5["GeoNameIdentifier"])
#We delete because the ID does not give us reference to the city
```

```
df_virus5["OsSuite"].value_counts()
```

```
768    311226
256    187950
272       677
16         53
400        47
305        33
784        10
274         2
18          1
Name: OsSuite, dtype: int64
```

IeVerIdentifier

```
df_virus5['IeVerIdentifier'].isnull().sum()
```

```
3209
```

```
df_virus5['IeVerIdentifier'].value_counts().head(30)
```

```
137.0    219141
117.0     98667
108.0     26353
111.0     26143
98.0      19731
135.0     12314
74.0      11395
53.0      11352
94.0       9715
105.0      9695
333.0      8751
107.0      7080
103.0      6364
96.0       4695
76.0       3242
71.0       2275
41.0       1918
114.0      1862
323.0      1814
335.0      1422
334.0      1356
87.0        936
81.0        809
78.0        655
73.0        615
82.0        571
337.0       540
42.0        502
302.0       413
85.0        399
Name: IeVerIdentifier, dtype: int64
```

```
# We will bring together all the records, except the 8 with the most values, in the "Others" category
```

```
df_virus5['IeVerIdentifier'] = setOthers(df_virus5, 'IeVerIdentifier', 8)
```

```
df_virus5['IeVerIdentifier'].value_counts().head()
```

```
137.0     219141
117.0      98667
Others     74903
108.0      26353
111.0      26143
Name: IeVerIdentifier, dtype: int64
```

UacLuaenable

```
df_virus5['UacLuaenable'].isnull().sum()
```

```
623
```

```
df_virus5['UacLuaenable'].value_counts()
```

```
1.0          496275
0.0            3086
48.0             13
2.0               1
6357062.0         1
Name: UacLuaenable, dtype: int64
```

```
# This variable does not add value, since it is almost univariate. We will delete it.
```

```
del(df_virus5["UacLuaenable"])
```

Census_OEMNameIdentifier

```
df_virus5['Census_OEMNameIdentifier'].value_counts()
```

```
2668.0    72011
2102.0    57924
1443.0    53210
2206.0    51888
585.0     50304
          ...
3997.0        1
3546.0        1
4037.0        1
2429.0        1
1900.0        1
Name: Census_OEMNameIdentifier, Length: 1589, dtype: int64
```

```
df_virus5['Census_OEMNameIdentifier'] = df_virus5['Census_OEMNameIdentifier'].astype(str).str.slice(stop=1)
```

```
df_virus5['Census_OEMNameIdentifier'].value_counts()
```

```
2    193963
5    113124
1     79148
4     67406
3     33095
n      5381
6      4066
7      2083
9      1531
8       202
Name: Census_OEMNameIdentifier, dtype: int64
```

Census_OEMModelIdentifier

```
df_virus['Census_OEMModelIdentifier'].value_counts()
```

```
313586.0    17092
242491.0    14726
317701.0     7676
317708.0     6541
228975.0     4389
            ...
1917.0          1
318232.0        1
342796.0        1
8067.0          1
35038.0         1
Name: Census_OEMModelIdentifier, Length: 40697, dtype: int64
```

```
df_virus5['Census_OEMModelIdentifier'] = df_virus5['Census_OEMModelIdentifier'].astype(str).str.slice(stop=1)
```

```
df_virus5['Census_OEMModelIdentifier'].value_counts()
```

```
2    228514
3    142484
1    114780
n      5764
4      2392
9      1818
7      1338
6      1190
8       906
```

```
5       813
Name: Census_OEMModelIdentifier, dtype: int64
```

Census_ProcessorCoreCount

```
df_virus5['Census_ProcessorCoreCount'].isnull().sum()
```

```
2347
```

```
df_virus5['Census_ProcessorCoreCount'].value_counts()
```

```
4.0     304102
2.0     129177
8.0      48995
12.0      5184
6.0       4023
1.0       3980
16.0      1006
3.0        752
32.0       113
20.0        97
24.0        95
40.0        39
28.0        17
36.0        16
48.0        15
5.0         10
10.0        10
56.0         8
7.0          3
11.0         2
64.0         2
52.0         1
44.0         1
88.0         1
80.0         1
14.0         1
46.0         1
Name: Census_ProcessorCoreCount, dtype: int64
```

```
df_virus5['Census_ProcessorCoreCount'] = setOthers(df_virus5, 'Census_ProcessorCoreCount', 3)
```

```
df_virus5['Census_ProcessorCoreCount'].value_counts()
```

```
4.0       304102
2.0       129177
8.0        48995
Others     17725
Name: Census_ProcessorCoreCount, dtype: int64
```

Census_ProcessorManufacturerIdentifier

```
df_virus5['Census_ProcessorManufacturerIdentifier'].isnull().sum()
```

```
2347
```

```
df_virus5['Census_ProcessorManufacturerIdentifier'].value_counts()
```

```
5.0     439028
1.0      58603
10.0        14
3.0          7
Name: Census_ProcessorManufacturerIdentifier, dtype: int64
```

```
df_virus5['Census_ProcessorManufacturerIdentifier'] = setOthers(df_virus5, 'Census_ProcessorManufacturerIdentifier', 2)
```

```
df_virus5['Census_ProcessorManufacturerIdentifier'].value_counts()
```

```
5.0       439028
1.0        58603
Others      2368
Name: Census_ProcessorManufacturerIdentifier, dtype: int64
```

Census_ProcessorModelIdentifier

```
df_virus5['Census_ProcessorModelIdentifier'].value_counts()
```

```
2697.0    16180
1998.0    14874
2660.0    10736
2373.0     9885
1992.0     9502
          ...
4091.0        1
1328.0        1
1852.0        1
3130.0        1
4027.0        1
Name: Census_ProcessorModelIdentifier, Length: 2243, dtype: int64
```

```
df_virus5['Census_ProcessorModelIdentifier'] = df_virus5['Census_ProcessorModelIdentifier'].astype(str).str.slice(stop=1)
```

```
df_virus5['Census_ProcessorModelIdentifier'].value_counts()
```

```
2    277303
3     99114
1     85616
6     15548
4     11902
8      5102
n      2349
7      1070
5      1039
9       956
Name: Census_ProcessorModelIdentifier, dtype: int64
```

Census_PrimaryDiskTotalCapacity

```
df_virus5['Census_PrimaryDiskTotalCapacity'].value_counts()
```

```
476940.0    158683
953869.0    122183
305245.0     26687
122104.0     26444
244198.0     25159
             ...
948333.0         1
20646.0          1
122069.0         1
190652.0         1
152499.0         1
Name: Census_PrimaryDiskTotalCapacity, Length: 1103, dtype: int64
```

```
df_virus5['Census_PrimaryDiskTotalCapacity'] = df_virus5['Census_PrimaryDiskTotalCapacity'].astype(str).str.slice(stop=1)
```

```
df_virus5['Census_PrimaryDiskTotalCapacity'].value_counts()
```

```
4    167846
9    124293
2     82240
1     61536
3     28918
7     16284
5      8821
6      6912
n      2976
8       173
Name: Census_PrimaryDiskTotalCapacity, dtype: int64
```

Census_SystemVolumeTotalCapacity

```
df_virus5['Census_SystemVolumeTotalCapacity'].isnull().sum()
```

```
2976
```

```
df_virus5['Census_SystemVolumeTotalCapacity'].value_counts()
```

```
28542.0     2922
926992.0    2866
476389.0    2380
476324.0    2306
102400.0    2303
            ...
221517.0       1
580181.0       1
127615.0       1
236484.0       1
```

```
470309.0       1
Name: Census_SystemVolumeTotalCapacity, Length: 142051, dtype: int64
```

```
df_virus5['Census_SystemVolumeTotalCapacity'] = df_virus5['Census_SystemVolumeTotalCapacity'].astype(str)
```

```
df_virus5['Census_SystemVolumeTotalCapacity'] = df_virus5['Census_SystemVolumeTotalCapacity'].str.slice(stop=2)
```

```
df_virus5['Census_SystemVolumeTotalCapacity'].value_counts()
```

```
47    43532
95    28539
46    28011
93    23949
45    22434
11    22313
12    21857
23    21015
10    20727
28    19874
24    19257
22    16519
29    15129
19    13072
15    12076
30    10474
43     9112
99     8160
94     7560
14     7192
20     6994
92     6672
18     5983
38     5449
58     5292
91     4503
48     4494
90     4202
49     3701
69     3625
21     3404
59     3373
25     3355
44     3125
17     2998
na     2976
13     2717
60     2685
50     2526
71     2481
27     2378
75     2119
70     2119
51     2084
42     2070
26     1959
81     1837
16     1820
68     1695
61     1630
98     1552
35     1543
79     1370
39     1369
40     1327
34     1258
76     1219
52      990
```

```
df_virus5['Census_SystemVolumeTotalCapacity'] = df_virus5['Census_SystemVolumeTotalCapacity'].str.slice(stop=1)
```

```
df_virus5['Census_SystemVolumeTotalCapacity'].value_counts()
```

```
4    118683
1    110755
2    109884
9     86622
3     23881
5     18126
6     12398
7     11127
8      5547
n      2976
Name: Census_SystemVolumeTotalCapacity, dtype: int64
```

```
df_virus5['Census_SystemVolumeTotalCapacity'] = setOthers(df_virus5, 'Census_SystemVolumeTotalCapacity', 4)
```

```
df_virus5['Census_SystemVolumeTotalCapacity'].value_counts()
```

```
4         118683
1         110755
2         109884
9          86622
Others     74055
Name: Census_SystemVolumeTotalCapacity, dtype: int64
```

Census_TotalPhysicalRAM

```
df_virus5['Census_TotalPhysicalRAM'].isnull().sum()
```

```
4556
```

```
df_virus5['Census_TotalPhysicalRAM'].value_counts().head(25)
```

```
4096.0      228677
8192.0      123174
2048.0       61694
16384.0      30179
6144.0       22349
12288.0       9074
3072.0        8468
1024.0        3603
32768.0       3308
24576.0        700
10240.0        596
5120.0         413
65536.0        349
1536.0         286
2560.0         259
20480.0        238
4095.0         237
2047.0         193
8191.0         110
14336.0         92
7168.0          80
131072.0        57
49152.0         57
3584.0          49
3071.0          44
Name: Census_TotalPhysicalRAM, dtype: int64
```

```
df_virus5['Census_TotalPhysicalRAM'] = setOthers(df_virus5, 'Census_TotalPhysicalRAM', 4)
```

```
df_virus5['Census_TotalPhysicalRAM'].value_counts().head(25)
```

```
4096.0     228677
8192.0     123174
2048.0      61694
Others      56275
16384.0     30179
Name: Census_TotalPhysicalRAM, dtype: int64
```

Census_InternalPrimaryDiagonalDisplaySizeInInches

```
df_virus5['Census_InternalPrimaryDiagonalDisplaySizeInInches'].value_counts()
```

```
15.5    171319
13.9     52873
14.0     30492
11.6     17592
21.5     15441
         ...
32.5         1
85.8         1
49.1         1
60.2         1
95.4         1
Name: Census_InternalPrimaryDiagonalDisplaySizeInInches, Length: 520, dtype: int64
```

```
df_virus5['Census_InternalPrimaryDiagonalDisplaySizeInInches'] = df_virus5['Census_InternalPrimaryDiagonalDisplaySizeInInches'].astype(st
```

```
df_virus5['Census_InternalPrimaryDiagonalDisplaySizeInInches'].value_counts()
```

```
1    412011
2     72665
```

```
3      3539
n      2654
8      2117
7      2108
4      1934
5      1501
9       862
6       608
Name: Census_InternalPrimaryDiagonalDisplaySizeInInches, dtype: int64
```

Census_InternalPrimaryDisplayResolutionHorizontal

```
df_virus5['Census_InternalPrimaryDisplayResolutionHorizontal'].isnull().sum()
```

```
2650
```

```
df_virus5['Census_InternalPrimaryDisplayResolutionHorizontal'].value_counts().head(20)
```

```
1366.0    251999
1920.0    125095
1280.0     29690
1600.0     28231
1024.0     19326
1440.0      9441
1360.0      7058
1680.0      6234
2560.0      3989
2736.0      3159
3840.0      2479
800.0       1784
2160.0      1760
3200.0       978
3000.0       606
2880.0       557
1368.0       480
640.0        464
1152.0       433
2048.0       308
Name: Census_InternalPrimaryDisplayResolutionHorizontal, dtype: int64
```

```
df_virus5['Census_InternalPrimaryDisplayResolutionHorizontal'] = setOthers(df_virus5, 'Census_InternalPrimaryDisplayResolutionHorizontal'
```

```
df_virus5['Census_InternalPrimaryDisplayResolutionHorizontal'].value_counts().head(20)
```

```
1366.0    251999
1920.0    125095
Others     45658
1280.0     29690
1600.0     28231
1024.0     19326
Name: Census_InternalPrimaryDisplayResolutionHorizontal, dtype: int64
```

Census_InternalPrimaryDisplayResolutionVertical

```
df_virus5['Census_InternalPrimaryDisplayResolutionVertical'].isnull().sum()
```

```
2650
```

```
df_virus5['Census_InternalPrimaryDisplayResolutionVertical'].value_counts().head(20)
```

```
768.0     277761
1080.0    121145
900.0      36912
800.0      14687
1024.0     10603
1050.0      6397
1440.0      4523
1200.0      4392
600.0       3492
1824.0      3157
720.0       2763
2160.0      2619
1280.0      1650
1800.0      1439
1600.0       914
2000.0       606
480.0        482
1920.0       478
864.0        418
960.0        312
Name: Census_InternalPrimaryDisplayResolutionVertical, dtype: int64
```

```
df_virus5['Census_InternalPrimaryDisplayResolutionVertical'] = setOthers(df_virus5, 'Census_InternalPrimaryDisplayResolutionVertical', 5)
```

```
df_virus5['Census_InternalPrimaryDisplayResolutionVertical'].value_counts().head(20)
```

```
768.0     277761
1080.0    121145
Others     38891
900.0      36912
800.0      14687
1024.0     10603
Name: Census_InternalPrimaryDisplayResolutionVertical, dtype: int64
```

Census_InternalBatteryNumberOfCharges

```
df_virus5['Census_InternalBatteryNumberOfCharges'].isnull().sum()
```

```
15038
```

```
df_virus['Census_InternalBatteryNumberOfCharges'].value_counts()
```

```
0.000000e+00    283189
4.294967e+09    126436
1.000000e+00      2955
1.600000e+01      1517
2.000000e+00      1503
                  ...
8.331000e+03         1
5.772700e+04         1
1.030000e+03         1
4.354400e+04         1
1.900000e+03         1
Name: Census_InternalBatteryNumberOfCharges, Length: 5248, dtype: int64
```

```
df_virus5['Census_InternalBatteryNumberOfCharges'].value_counts()
```

```
0.000000e+00    283189
4.294967e+09    126435
1.000000e+00      2955
1.600000e+01      1517
2.000000e+00      1503
                  ...
8.331000e+03         1
5.772700e+04         1
1.030000e+03         1
4.354400e+04         1
1.900000e+03         1
Name: Census_InternalBatteryNumberOfCharges, Length: 5248, dtype: int64
```

```
len(df_virus5[df_virus5['Census_InternalBatteryNumberOfCharges'] == 4294967295])
```

```
126435
```

```
filtro_internalbattery = df_virus5['Census_InternalBatteryNumberOfCharges'] == 4294967295
```

```
df_virus5[filtro_internalbattery]["HasDetections"].mean()
```

```
0.5157432672914937
```

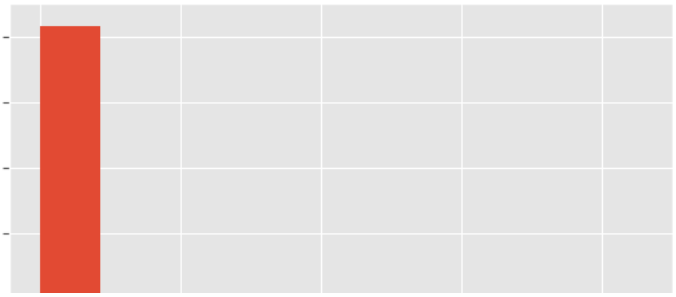
```
filtro_internalbattery_2 = df_virus5['Census_InternalBatteryNumberOfCharges'] == 0
```

```
df_virus5[filtro_internalbattery_2]["HasDetections"].mean()
```

```
0.4970955792774437
```

```
df_virus5['Census_InternalBatteryNumberOfCharges'].hist()
```

```
<Axes: >
```




```
#We proceed to eliminate the column since:
 # Has nulls
 #40% are 0.0 values
 #20% has value 4.x
 # The information we have about this variable is N/A
 # The average of each of the variables with the greatest amount of data represents almost the same % of the TARGET
```

```
del(df_virus5["Census_InternalBatteryNumberOfCharges"])
```

1e9

Census_OsBuildNumber

```
df_virus5['Census_OSBuildNumber'].value_counts()
```

```
14393     43702
10586     33193
10240     15106
17738       165
17692       146
17744       135
17758        99
17746        65
17754        59
17763        59
17751        54
17741        45
17755        40
17735        39
17733        35
17686        34
17760        31
17133        21
17682        20
17677        17
17672        15
18237        14
18234        14
18242        13
17661         9
17713         8
17666         7
17634         4
17639         3
17747         3
17730         2
17655         2
14986         2
17650         2
17046         1
17004         1
14295         1
18219         1
17623         1
17749         1
14421         1
14971         1
17723         1
16193         1
14926         1
17604         1
17618         1
11082         1
14328         1
15019         1
18214         1
16251         1
14946         1
9600          1
17074         1
17753         1
15042         1
15048         1
Name: Census_OSBuildNumber, dtype: int64
```

```
df_virus5['Census_OSBuildNumber'] = df_virus5['Census_OSBuildNumber'].astype(str).str.slice(stop=2)
```

```
df_virus5['Census_OSBuildNumber'].value_counts()
```

```
17    227106
16    136574
10     48299
15     44265
14     43710
18        43
96         1
11         1
Name: Census_OSBuildNumber, dtype: int64
```

Census_OSBuildRevision

```
df_virus5['Census_OSBuildRevision'].value_counts()
```

```
228      79975
165      50511
431      30519
285      26289
112      19503
         ...
17643        1
1230         1
491          1
206          1
16399        1
Name: Census_OSBuildRevision, Length: 242, dtype: int64
```

```
df_virus5['Census_OSBuildRevision'] = df_virus5['Census_OSBuildRevision'].astype(str).str.slice(stop=1)
```

```
df_virus5['Census_OSBuildRevision'].value_counts()
```

```
1    190618
2    148272
4     52819
3     32936
5     26249
6     23725
0      9175
8      5552
7      5465
9      5188
Name: Census_OSBuildRevision, dtype: int64
```

Census_OSInstallLanguageIdentifier

```
df_virus5['Census_OSInstallLanguageIdentifier'].value_counts()
```

```
8.0     178405
9.0      58012
7.0      28766
29.0     27669
14.0     24191
37.0     22635
10.0     20268
26.0     18520
5.0      14085
35.0     11497
39.0     11221
18.0     10600
20.0      9512
24.0      8127
25.0      7502
27.0      5956
19.0      4775
17.0      4695
1.0       4516
3.0       4067
6.0       2881
33.0      2807
15.0      2311
4.0       1996
30.0      1793
23.0      1652
31.0      1213
12.0      1119
2.0       1096
36.0       846
16.0       837
```

```
28.0       713
34.0       584
13.0       581
21.0       413
32.0       246
11.0       197
38.0       185
22.0       178
Name: Census_OSInstallLanguageIdentifier, dtype: int64
```

```
df_virus5['Census_OSInstallLanguageIdentifier'] = df_virus5['Census_OSInstallLanguageIdentifier'].astype(str).str.slice(stop=1)
```

```
df_virus5['Census_OSInstallLanguageIdentifier'].value_counts()
```

```
8    178405
2     81338
1     74090
9     58012
3     57094
7     28766
5     14085
n      3332
6      2881
4      1996
Name: Census_OSInstallLanguageIdentifier, dtype: int64
```

Census_OSUILocaleIdentifier

```
df_virus5['Census_OSUILocaleIdentifier'].value_counts()
```

```
151       863
62        857
123       718
48        597
145       589
90        432
156       421
68        259
136       256
38        206
159       195
92        188
17        126
42         17
15         15
32         15
126        13
54         11
161        10
59          9
36          7
12          6
131         5
93          5
63          5
37          5
152         4
```

```
31          1
70          1
89          1
```

```
df_virus5['Census_OSUILocaleIdentifier'] = df_virus5['Census_OSUILocaleIdentifier'].astype(str).str.slice(stop=1)
```

```
df_virus5["Census_OSUILocaleIdentifier"].value_counts()
```

```
3    286212
1    126014
4     26157
2     23090
7     15464
8      9599
5      6959
6      5873
9       631
Name: Census_OSUILocaleIdentifier, dtype: int64
```

Census_FirmwareManufacturerIdentifier

```
df_virus5["Census_FirmwareManufacturerIdentifier"].value_counts()
```

```
142.0    151373
628.0     68781
554.0     65673
355.0     52758
556.0     44919
          ...
33.0          1
697.0         1
13.0          1
334.0         1
585.0         1
Name: Census_FirmwareManufacturerIdentifier, Length: 303, dtype: int64
```

```
df_virus5['Census_FirmwareManufacturerIdentifier'] = df_virus5['Census_FirmwareManufacturerIdentifier'].astype(str).str.slice(stop=1)
```

```
df_virus5["Census_FirmwareManufacturerIdentifier"].value_counts()
```

```
1    167013
5    148424
6     77434
3     53609
8     20483
9     10733
n     10349
4      9638
7      1935
2       381
Name: Census_FirmwareManufacturerIdentifier, dtype: int64
```

Census_FirmwareVersionIdentifier

```
df_virus5["Census_FirmwareVersionIdentifier"].value_counts()
```

```
33105.0    5036
33111.0    3356
33054.0    3124
33108.0    3071
63175.0    2968
           ...
41482.0       1
41752.0       1
20762.0       1
15577.0       1
54497.0       1
Name: Census_FirmwareVersionIdentifier, Length: 23569, dtype: int64
```

```
df_virus5['Census_FirmwareVersionIdentifier'] = df_virus5['Census_FirmwareVersionIdentifier'].astype(str).str.slice(stop=1)
```

```
df_virus5["Census_FirmwareVersionIdentifier"].value_counts()
```

```
3    130590
1     90024
6     88402
5     41183
7     40024
2     39466
4     33931
9     15390
```

```
8     11928
n      9061
Name: Census_FirmwareVersionIdentifier, dtype: int64
```

Unnamed: 0

```
del(df_virus5["Unnamed: 0"]) #We eliminate because it has no value
```

Census_IsWIMBootEnabled

```
df_virus5['Census_IsWIMBootEnabled'].value_counts()
```

```
0.0    182333
Name: Census_IsWIMBootEnabled, dtype: int64
```

```
del(df_virus5["Census_IsWIMBootEnabled"]) #We eliminate because it has too many nulls and those that are not are all 0
```

```
df_virus5.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 499999 entries, 0 to 499999
Data columns (total 64 columns):
 #   Column                                        Non-Null Count   Dtype
---  ------                                        --------------   -----
 0   ProductName                                   499999 non-null  object
 1   EngineVersion                                 499999 non-null  category
 2   AppVersion                                    499999 non-null  object
 3   AvSigVersion                                  499999 non-null  category
 4   RtpStateBitfield                              498167 non-null  float64
 5   IsSxsPassiveMode                              499999 non-null  int64
 6   AVProductStatesIdentifier                     499999 non-null  category
 7   AVProductsInstalled                           499999 non-null  category
 8   AVProductsEnabled                             499999 non-null  category
 9   HasTpm                                        499999 non-null  int64
 10  Platform                                      499999 non-null  object
 11  Processor                                     499999 non-null  object
 12  OsVer                                         499999 non-null  object
 13  OsBuild                                       499999 non-null  object
 14  OsSuite                                       499999 non-null  object
 15  OsPlatformSubRelease                          499999 non-null  object
 16  OsBuildLab                                    499999 non-null  object
 17  SkuEdition                                    499999 non-null  category
 18  IsProtected                                   499999 non-null  object
 19  SMode                                         499999 non-null  object
 20  IeVerIdentifier                               499999 non-null  category
 21  SmartScreen                                   499999 non-null  category
 22  Firewall                                      499999 non-null  object
 23  Census_MDC2FormFactor                         499999 non-null  category
 24  Census_OEMNameIdentifier                      499999 non-null  object
 25  Census_OEMModelIdentifier                     499999 non-null  object
 26  Census_ProcessorCoreCount                     499999 non-null  category
 27  Census_ProcessorManufacturerIdentifier        499999 non-null  category
 28  Census_ProcessorModelIdentifier               499999 non-null  object
 29  Census_PrimaryDiskTotalCapacity               499999 non-null  object
 30  Census_PrimaryDiskTypeName                    499999 non-null  category
 31  Census_SystemVolumeTotalCapacity              499999 non-null  category
 32  Census_HasOpticalDiskDrive                    499999 non-null  int64
 33  Census_TotalPhysicalRAM                       499999 non-null  category
 34  Census_ChassisTypeName                        499999 non-null  category
 35  Census_InternalPrimaryDiagonalDisplaySizeInInches  499999 non-null  object
 36  Census_InternalPrimaryDisplayResolutionHorizontal  499999 non-null  category
 37  Census_InternalPrimaryDisplayResolutionVertical    499999 non-null  category
 38  Census_PowerPlatformRoleName                  499999 non-null  category
 39  Census_OSVersion                              499999 non-null  object
 40  Census_OSArchitecture                         499999 non-null  object
 41  Census_OSBranch                               499999 non-null  category
 42  Census_OSBuildNumber                          499999 non-null  object
 43  Census_OSBuildRevision                        499999 non-null  object
 44  Census_OSEdition                              499999 non-null  category
 45  Census_OSInstallTypeName                      499999 non-null  object
 46  Census_OSInstallLanguageIdentifier            499999 non-null  object
 47  Census_OSUILocaleIdentifier                   499999 non-null  object
 48  Census_OSWUAutoUpdateOptionsName              499999 non-null  object
 49  Census_IsPortableOperatingSystem              499999 non-null  int64
 50  Census_GenuineStateName                       499999 non-null  object
 51  Census_ActivationChannel                      499999 non-null  object
 52  Census_IsFlightsDisabled                      499999 non-null  object
```

RtpStateBitfield

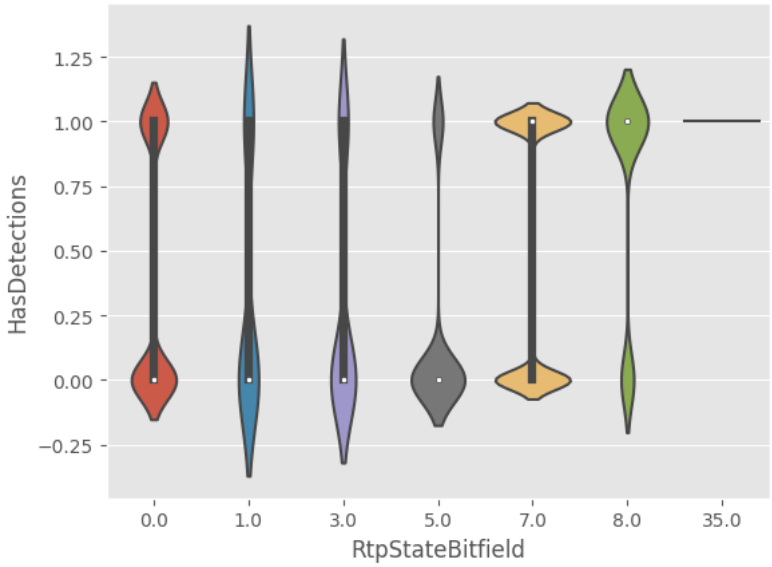
```
df_virus6 = df_virus5.copy()
```

```
df_virus6["RtpStateBitfield"].value_counts()
```

```
7.0     484840
0.0      10610
8.0       1277
5.0       1158
3.0        176
1.0        105
35.0         1
Name: RtpStateBitfield, dtype: int64
```

```
sns.violinplot(x="RtpStateBitfield", y=TARGET, data=df_virus6)
```

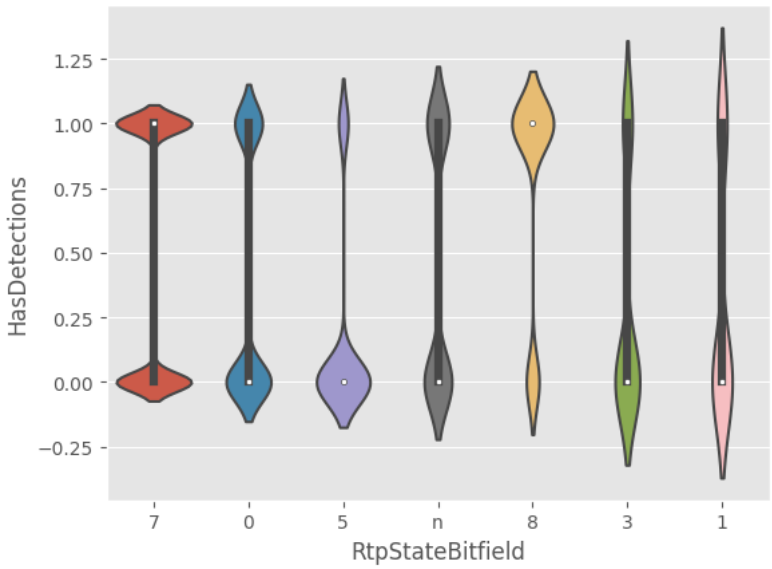
```
<Axes: xlabel='RtpStateBitfield', ylabel='HasDetections'>
```




```
df_virus6['RtpStateBitfield'] = df_virus6['RtpStateBitfield'].astype(str).str.slice(stop=1)
```

```
sns.violinplot(x="RtpStateBitfield", y=TARGET, data=df_virus6)
```

```
<Axes: xlabel='RtpStateBitfield', ylabel='HasDetections'>
```



IsSxsPassiveMode

```
df_virus6["IsSxsPassiveMode"].value_counts()
#OHE
```

```
0    491378
1      8621
Name: IsSxsPassiveMode, dtype: int64
```

HasTpm

```
df_virus6["HasTpm"].value_counts()
#OHE
```

```
1    493907
0      6092
Name: HasTpm, dtype: int64
```

Census_HasOpticalDiskDrive

```
df_virus6["Census_HasOpticalDiskDrive"].value_counts()
#OHE
```

```
0    461482
1     38517
Name: Census_HasOpticalDiskDrive, dtype: int64
```

Census_IsTouchEnabled

```
df_virus6["Census_IsTouchEnabled"].value_counts()
#OHE
```

```
0    437282
1     62717
Name: Census_IsTouchEnabled, dtype: int64
```

Wdft_RegionIdentifier

```
df_virus6["Wdft_RegionIdentifier"].value_counts()
```

```
10.0    100391
11.0     75612
3.0      73069
1.0      68692
15.0     57346
7.0      33362
8.0      15761
13.0     12681
5.0      11466
12.0      9115
6.0       8750
4.0       7586
9.0       4521
2.0       4470
14.0       227
Name: Wdft_RegionIdentifier, dtype: int64
```

```
df_virus6['Wdft_RegionIdentifier'] = setOthers(df_virus6, 'Wdft_RegionIdentifier', 9)
```

```
df_virus6["Wdft_RegionIdentifier"].value_counts()
```

```
10.0      100391
11.0       75612
3.0        73069
1.0        68692
15.0       57346
Others     51619
7.0        33362
8.0        15761
13.0       12681
5.0        11466
Name: Wdft_RegionIdentifier, dtype: int64
```

Correlation

```
df_virus6.corr().style.background_gradient(cmap="coolwarm")
```

```
<ipython-input-373-181ada5d9345>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future v
  df_virus6.corr().style.background_gradient(cmap="coolwarm")
```

	IsSxsPassiveMode	HasTpm	Census_HasOpticalDiskDrive	Census_IsPortableOperatingSystem	Census_
IsSxsPassiveMode	1.000000	0.013870	0.010481	-0.001009	
HasTpm	0.013870	1.000000	0.005763	-0.053331	
Census_HasOpticalDiskDrive	0.010481	0.005763	1.000000	0.001935	
Census_IsPortableOperatingSystem	-0.001009	-0.053331	0.001935	1.000000	
Census_IsSecureBootEnabled	0.020659	0.087262	-0.060760	-0.013087	

```
df_virus6.isnull().sum()
```

```
AVProductStatesIdentifier                          0
AVProductsInstalled                                0
AVProductsEnabled                                  0
HasTpm                                             0
Platform                                           0
Processor                                          0
OsVer                                              0
OsBuild                                            0
OsSuite                                            0
OsPlatformSubRelease                               0
OsBuildLab                                         0
SkuEdition                                         0
IsProtected                                        0
SMode                                              0
IeVerIdentifier                                    0
SmartScreen                                        0
Firewall                                           0
Census_MDC2FormFactor                              0
Census_OEMNameIdentifier                           0
Census_OEMModelIdentifier                          0
Census_ProcessorCoreCount                          0
Census_ProcessorManufacturerIdentifier             0
Census_ProcessorModelIdentifier                    0
Census_PrimaryDiskTotalCapacity                    0
Census_PrimaryDiskTypeName                         0
Census_SystemVolumeTotalCapacity                   0
Census_HasOpticalDiskDrive                         0
Census_TotalPhysicalRAM                            0
Census_ChassisTypeName                             0
Census_InternalPrimaryDiagonalDisplaySizeInInches  0
Census_InternalPrimaryDisplayResolutionHorizontal  0
Census_InternalPrimaryDisplayResolutionVertical    0
Census_PowerPlatformRoleName                       0
Census_OSVersion                                   0
Census_OSArchitecture                              0
Census_OSBranch                                    0
Census_OSBuildNumber                               0
Census_OSBuildRevision                             0
Census_OSEdition                                   0
Census_OSInstallTypeName                           0
Census_OSInstallLanguageIdentifier                 0
Census_OSUILocaleIdentifier                        0
Census_OSWUAutoUpdateOptionsName                   0
Census_IsPortableOperatingSystem                   0
Census_GenuineStateName                            0
Census_ActivationChannel                           0
Census_IsFlightsDisabled                           0
Census_FlightRing                                  0
Census_FirmwareManufacturerIdentifier              0
Census_FirmwareVersionIdentifier                   0
Census_IsSecureBootEnabled                         0
Census_IsVirtualDevice                             0
Census_IsTouchEnabled                              0
Census_IsPenCapable                                0
Census_IsAlwaysOnAlwaysConnectedCapable            0
Wdft_IsGamer                                       0
Wdft_RegionIdentifier                              0
HasDetections                                      0
```

```
df_virus6.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 499999 entries, 0 to 499999
Data columns (total 64 columns):
 #   Column                                     Non-Null Count   Dtype
---  ------                                     --------------   -----
 0   ProductName                                499999 non-null  object
 1   EngineVersion                              499999 non-null  category
 2   AppVersion                                 499999 non-null  object
 3   AvSigVersion                               499999 non-null  category
 4   RtpStateBitfield                           499999 non-null  object
 5   IsSxsPassiveMode                           499999 non-null  int64
 6   AVProductStatesIdentifier                  499999 non-null  category
```

```
 7   AVProductsInstalled                             499999 non-null  category
 8   AVProductsEnabled                               499999 non-null  category
 9   HasTpm                                          499999 non-null  int64
 10  Platform                                        499999 non-null  object
 11  Processor                                       499999 non-null  object
 12  OsVer                                           499999 non-null  object
 13  OsBuild                                         499999 non-null  object
 14  OsSuite                                         499999 non-null  object
 15  OsPlatformSubRelease                            499999 non-null  object
 16  OsBuildLab                                      499999 non-null  object
 17  SkuEdition                                      499999 non-null  category
 18  IsProtected                                     499999 non-null  object
 19  SMode                                           499999 non-null  object
 20  IeVerIdentifier                                 499999 non-null  category
 21  SmartScreen                                     499999 non-null  category
 22  Firewall                                        499999 non-null  object
 23  Census_MDC2FormFactor                           499999 non-null  category
 24  Census_OEMNameIdentifier                        499999 non-null  object
 25  Census_OEMModelIdentifier                       499999 non-null  object
 26  Census_ProcessorCoreCount                       499999 non-null  category
 27  Census_ProcessorManufacturerIdentifier          499999 non-null  category
 28  Census_ProcessorModelIdentifier                 499999 non-null  object
 29  Census_PrimaryDiskTotalCapacity                 499999 non-null  object
 30  Census_PrimaryDiskTypeName                      499999 non-null  category
 31  Census_SystemVolumeTotalCapacity                499999 non-null  category
 32  Census_HasOpticalDiskDrive                      499999 non-null  int64
 33  Census_TotalPhysicalRAM                         499999 non-null  category
 34  Census_ChassisTypeName                          499999 non-null  category
 35  Census_InternalPrimaryDiagonalDisplaySizeInInches  499999 non-null  object
 36  Census_InternalPrimaryDisplayResolutionHorizontal  499999 non-null  category
 37  Census_InternalPrimaryDisplayResolutionVertical    499999 non-null  category
 38  Census_PowerPlatformRoleName                    499999 non-null  category
 39  Census_OSVersion                                499999 non-null  object
 40  Census_OSArchitecture                           499999 non-null  object
 41  Census_OSBranch                                 499999 non-null  category
 42  Census_OSBuildNumber                            499999 non-null  object
 43  Census_OSBuildRevision                          499999 non-null  object
 44  Census_OSEdition                                499999 non-null  category
 45  Census_OSInstallTypeName                        499999 non-null  object
 46  Census_OSInstallLanguageIdentifier              499999 non-null  object
 47  Census_OSUILocaleIdentifier                     499999 non-null  object
 48  Census_OSWUAutoUpdateOptionsName                499999 non-null  object
 49  Census_IsPortableOperatingSystem                499999 non-null  int64
 50  Census_GenuineStateName                         499999 non-null  object
 51  Census_ActivationChannel                        499999 non-null  object
 52  Census_IsFlightsDisabled                        499999 non-null  object
```

```
df_virus6["Wdft_RegionIdentifier"].value_counts()
```

```
10.0      100391
11.0       75612
3.0        73069
1.0        68692
15.0       57346
Others     51619
7.0        33362
8.0        15761
13.0       12681
5.0        11466
Name: Wdft_RegionIdentifier, dtype: int64
```

```
columnas_cambio_acateg2 = ["SkuEdition", "IeVerIdentifier", "SmartScreen", "Census_MDC2FormFactor", "Census_ProcessorCoreCount", "Census_
                           "Census_TotalPhysicalRAM", "Census_ChassisTypeName", "Census_InternalPrimaryDisplayResolutionHorizontal", "Cen
                           "Census_OSBranch", "Census_OSEdition", "Census_FlightRing", "Wdft_RegionIdentifier", "EngineVersion", "AvSigVe
```

```
for column in columnas_cambio_acateg2:
  df_virus6[column] = df_virus6[column].astype('object')
```

```
df_virus6.info()
```

```
 6   AVProductStatesIdentifier                       499999 non-null  object
 7   AVProductsInstalled                             499999 non-null  object
 8   AVProductsEnabled                               499999 non-null  object
 9   HasTpm                                          499999 non-null  object
 10  Platform                                        499999 non-null  object
```

```
 22  Firewall                                          499999 non-null  object
 23  Census_MDC2FormFactor                             499999 non-null  object
 24  Census_OEMNameIdentifier                          499999 non-null  object
 25  Census_OEMModelIdentifier                         499999 non-null  object
 26  Census_ProcessorCoreCount                         499999 non-null  object
 27  Census_ProcessorManufacturerIdentifier            499999 non-null  object
 28  Census_ProcessorModelIdentifier                   499999 non-null  object
 29  Census_PrimaryDiskTotalCapacity                   499999 non-null  object
 30  Census_PrimaryDiskTypeName                        499999 non-null  object
 31  Census_SystemVolumeTotalCapacity                  499999 non-null  object
 32  Census_HasOpticalDiskDrive                        499999 non-null  object
 33  Census_TotalPhysicalRAM                           499999 non-null  object
 34  Census_ChassisTypeName                            499999 non-null  object
 35  Census_InternalPrimaryDiagonalDisplaySizeInInches 499999 non-null  object
 36  Census_InternalPrimaryDisplayResolutionHorizontal 499999 non-null  object
 37  Census_InternalPrimaryDisplayResolutionVertical   499999 non-null  object
 38  Census_PowerPlatformRoleName                      499999 non-null  object
 39  Census_OSVersion                                  499999 non-null  object
 40  Census_OSArchitecture                             499999 non-null  object
 41  Census_OSBranch                                   499999 non-null  object
 42  Census_OSBuildNumber                              499999 non-null  object
 43  Census_OSBuildRevision                            499999 non-null  object
 44  Census_OSEdition                                  499999 non-null  object
 45  Census_OSInstallTypeName                          499999 non-null  object
 46  Census_OSInstallLanguageIdentifier                499999 non-null  object
 47  Census_OSUILocaleIdentifier                       499999 non-null  object
 48  Census_OSWUAutoUpdateOptionsName                  499999 non-null  object
 49  Census_IsPortableOperatingSystem                  499999 non-null  object
 50  Census_GenuineStateName                           499999 non-null  object
 51  Census_ActivationChannel                          499999 non-null  object
 52  Census_IsFlightsDisabled                          499999 non-null  object
 53  Census_FlightRing                                 499999 non-null  object
 54  Census_FirmwareManufacturerIdentifier             499999 non-null  object
 55  Census_FirmwareVersionIdentifier                  499999 non-null  object
 56  Census_IsSecureBootEnabled                        499999 non-null  object
 57  Census_IsVirtualDevice                            499999 non-null  object
 58  Census_IsTouchEnabled                             499999 non-null  object
 59  Census_IsPenCapable                               499999 non-null  object
 60  Census_IsAlwaysOnAlwaysConnectedCapable           499999 non-null  object
 61  Wdft_IsGamer                                      499999 non-null  object
 62  Wdft_RegionIdentifier                             499999 non-null  object
```

ONE HOT ENCODING

```
df_virus7 = df_virus6.copy()
```

```
df_virus7.info()
```

```
 6   AVProductStatesIdentifier                         499999 non-null  object
 7   AVProductsInstalled                               499999 non-null  object
 8   AVProductsEnabled                                 499999 non-null  object
 9   HasTpm                                            499999 non-null  object
 10  Platform                                          499999 non-null  object
 11  Processor                                         499999 non-null  object
 12  OsVer                                             499999 non-null  object
 13  OsBuild                                           499999 non-null  object
 14  OsSuite                                           499999 non-null  object
 15  OsPlatformSubRelease                              499999 non-null  object
 16  OsBuildLab                                        499999 non-null  object
 17  SkuEdition                                        499999 non-null  object
 18  IsProtected                                       499999 non-null  object
 19  SMode                                             499999 non-null  object
 20  IeVerIdentifier                                   499999 non-null  object
 21  SmartScreen                                       499999 non-null  object
 22  Firewall                                          499999 non-null  object
 23  Census_MDC2FormFactor                             499999 non-null  object
 24  Census_OEMNameIdentifier                          499999 non-null  object
 25  Census_OEMModelIdentifier                         499999 non-null  object
 26  Census_ProcessorCoreCount                         499999 non-null  object
 27  Census_ProcessorManufacturerIdentifier            499999 non-null  object
 28  Census_ProcessorModelIdentifier                   499999 non-null  object
 29  Census_PrimaryDiskTotalCapacity                   499999 non-null  object
 30  Census_PrimaryDiskTypeName                        499999 non-null  object
 31  Census_SystemVolumeTotalCapacity                  499999 non-null  object
 32  Census_HasOpticalDiskDrive                        499999 non-null  object
```

```
 44  Census_OSEdition                                 499999 non-null  object
 45  Census_OSInstallTypeName                         499999 non-null  object
 46  Census_OSInstallLanguageIdentifier               499999 non-null  object
 47  Census_OSUILocaleIdentifier                      499999 non-null  object
 48  Census_OSWUAutoUpdateOptionsName                 499999 non-null  object
 49  Census_IsPortableOperatingSystem                 499999 non-null  object
 50  Census_GenuineStateName                          499999 non-null  object
 51  Census_ActivationChannel                         499999 non-null  object
 52  Census_IsFlightsDisabled                         499999 non-null  object
 53  Census_FlightRing                                499999 non-null  object
 54  Census_FirmwareManufacturerIdentifier            499999 non-null  object
 55  Census_FirmwareVersionIdentifier                 499999 non-null  object
 56  Census_IsSecureBootEnabled                       499999 non-null  object
 57  Census_IsVirtualDevice                           499999 non-null  object
 58  Census_IsTouchEnabled                            499999 non-null  object
 59  Census_IsPenCapable                              499999 non-null  object
 60  Census_IsAlwaysOnAlwaysConnectedCapable          499999 non-null  object
 61  Wdft_IsGamer                                     499999 non-null  object
 62  Wdft_RegionIdentifier                            499999 non-null  object
 63  HasDetections                                    499999 non-null  int64
```

```
def OHE(df_virus7, column_name):
    _dummy_dataset = pd.get_dummies(df_virus7[column_name], prefix=column_name)
    df_virus7 = pd.concat([df_virus7, _dummy_dataset], axis=1)
    return df_virus7.drop(column_name, axis=1)
```

```
for column in df_virus7.select_dtypes(include=np.object).columns:
    df_virus7 = OHE(df_virus7, column)
```

```
<ipython-input-383-2a88a90f665a>:1: DeprecationWarning: `np.object` is a deprecated alias for the builtin `object`. To silence this
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  for column in df_virus7.select_dtypes(include=np.object).columns:
<ipython-input-382-013595376c30>:2: FutureWarning: In a future version, the Index constructor will not infer numeric dtypes when pa
  _dummy_dataset = pd.get_dummies(df_virus7[column_name], prefix=column_name)
<ipython-input-382-013595376c30>:2: FutureWarning: In a future version, the Index constructor will not infer numeric dtypes when pa
  _dummy_dataset = pd.get_dummies(df_virus7[column_name], prefix=column_name)
<ipython-input-382-013595376c30>:2: FutureWarning: In a future version, the Index constructor will not infer numeric dtypes when pa
  _dummy_dataset = pd.get_dummies(df_virus7[column_name], prefix=column_name)
<ipython-input-382-013595376c30>:2: FutureWarning: In a future version, the Index constructor will not infer numeric dtypes when pa
  _dummy_dataset = pd.get_dummies(df_virus7[column_name], prefix=column_name)
<ipython-input-382-013595376c30>:2: FutureWarning: In a future version, the Index constructor will not infer numeric dtypes when pa
  _dummy_dataset = pd.get_dummies(df_virus7[column_name], prefix=column_name)
<ipython-input-382-013595376c30>:2: FutureWarning: In a future version, the Index constructor will not infer numeric dtypes when pa
  _dummy_dataset = pd.get_dummies(df_virus7[column_name], prefix=column_name)
<ipython-input-382-013595376c30>:2: FutureWarning: In a future version, the Index constructor will not infer numeric dtypes when pa
  _dummy_dataset = pd.get_dummies(df_virus7[column_name], prefix=column_name)
<ipython-input-382-013595376c30>:2: FutureWarning: In a future version, the Index constructor will not infer numeric dtypes when pa
  _dummy_dataset = pd.get_dummies(df_virus7[column_name], prefix=column_name)
```

We check the variables that were previously boolean

```
bool = df_virus7.nunique() == 2
bool_columns = df_virus7.columns[bool].tolist()
bool_columns_df = df_virus7[bool_columns]
```

Download the variables from df_virus7 to review the variables created from boolean variables

```
df_virus7_description = df_virus7.describe(include='all').T
```

```
df_virus7_description.to_csv('df7_description.csv')
```

```
files.download('df7_description.csv')
```

```
# Of the variables created from booleans, we will eliminate 1 out of 2
```

```
del(df_virus7["ProductName_win8defender"])
del(df_virus7["IsSxsPassiveMode_1"])
del(df_virus7["AVProductsEnabled_Others"])
del(df_virus7["HasTpm_1"])
del(df_virus7["Platform_windows10"])
del(df_virus7["Processor_x86"])
```

```
del(df_virus7["OsVer_6"])
del(df_virus7["Census_HasOpticalDiskDrive_1"])
del(df_virus7["Census_OSArchitecture_x86"])
del(df_virus7["Census_IsPortableOperatingSystem_1"])
del(df_virus7["Census_IsTouchEnabled_1"])
del(df_virus7["Census_IsPenCapable_1"])
del(df_virus7["Census_IsSecureBootEnabled_1"])
```

MODELING

1. Importamos las librerias

```
from sklearn import model_selection # model assesment and model selection strategies
from sklearn import metrics # model evaluation metrics
import numpy as np
```

2. Partición del dataset - Validación

```
df_virus8 = df_virus7.copy()
```

```
#By not having a temporary variable, we decided to use randomness to partition the model

p_dev = 0.70 # Train´s %

df_virus8['is_train'] = np.random.uniform(0, 1, len(df_virus8)) <= p_dev
dev_df_virus8, val_df_virus8 = df_virus8[df_virus8['is_train']==True], df_virus8[df_virus8['is_train']==False]
df_virus8 = df_virus8.drop('is_train', 1)

print("Ejemplos usados para entrenar: ", len(dev_df_virus8))
print("Ejemplos usados para validación: ", len(val_df_virus8))
```

```
<ipython-input-392-5821099ed39f>:7: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the arg
  df_virus8 = df_virus8.drop('is_train', 1)
Ejemplos usados para entrenar:  349639
Ejemplos usados para validación:  150360
```

2.1 Asignación de atributos y target a las variables X e Y

```
dev_df_virus8_X = dev_df_virus8.drop('HasDetections', axis=1)
dev_df_virus8_y = dev_df_virus8[['HasDetections']]
```

```
val_df_virus8_X = val_df_virus8.drop('HasDetections', axis=1)
val_df_virus8_y = val_df_virus8[['HasDetections']]
```

2.2 Random hold out

```
X_train, X_test, y_train, y_test = model_selection.train_test_split(
                                       dev_df_virus8_X, # X
                                       dev_df_virus8_y, # y
                                       test_size = 0.30, # Size of aleatory split
                                       random_state = 42
                                   )
```

2.3 Checks previos

```
dev_df_virus8_X.head().T
```

	0	1	3	4	5
ProductName_mse	0	0	0	0	0
EngineVersion_1.1.14	0	0	0	0	0
EngineVersion_1.1.15	1	1	1	1	1
EngineVersion_Others	0	0	0	0	0
AppVersion_4.14	0	0	0	0	0

```
X_train.info(verbose=False)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 244747 entries, 258527 to 174435
Columns: 313 entries, ProductName_mse to is_train
dtypes: bool(1), uint8(312)
memory usage: 74.9 MB
```

Wdft_RegionIdentifier_Others	0	0	0	0	0

```
X_test.info(verbose=False)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 104892 entries, 245223 to 189560
Columns: 313 entries, ProductName_mse to is_train
dtypes: bool(1), uint8(312)
memory usage: 32.1 MB
```

```
X_train.describe().T.head()
```

	count	mean	std	min	25%	50%	75%	max
ProductName_mse	244747.0	0.010803	0.103375	0.0	0.0	0.0	0.0	1.0
EngineVersion_1.1.14	244747.0	0.088671	0.284269	0.0	0.0	0.0	0.0	1.0
EngineVersion_1.1.15	244747.0	0.895492	0.305919	0.0	1.0	1.0	1.0	1.0
EngineVersion_Others	244747.0	0.015837	0.124844	0.0	0.0	0.0	0.0	1.0
AppVersion_4.14	244747.0	0.027559	0.163706	0.0	0.0	0.0	0.0	1.0

```
X_test.describe().T.head()
```

	count	mean	std	min	25%	50%	75%	max
ProductName_mse	104892.0	0.010725	0.103007	0.0	0.0	0.0	0.0	1.0
EngineVersion_1.1.14	104892.0	0.089073	0.284850	0.0	0.0	0.0	0.0	1.0
EngineVersion_1.1.15	104892.0	0.894625	0.307038	0.0	1.0	1.0	1.0	1.0
EngineVersion_Others	104892.0	0.016302	0.126637	0.0	0.0	0.0	0.0	1.0
AppVersion_4.14	104892.0	0.026894	0.161775	0.0	0.0	0.0	0.0	1.0

```
y_train.describe().T.head()
```

	count	mean	std	min	25%	50%	75%	max
HasDetections	244747.0	0.500198	0.500001	0.0	0.0	1.0	1.0	1.0

```
y_test.describe().T.head()
```

	count	mean	std	min	25%	50%	75%	max
HasDetections	104892.0	0.500419	0.500002	0.0	0.0	1.0	1.0	1.0

Model definition

1. Importing libreries

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import export_graphviz
import graphviz
import pydotplus

!conda install python-graphviz -y
!conda install pydot -y
```

```
/bin/bash: conda: command not found
/bin/bash: conda: command not found
```

2. We instantiate the model

```
dt = DecisionTreeClassifier(
                        # criterion='gini',
                        # splitter='best',
                        max_depth=4,      # Importante, regulará el sobreajuste
                        # min_samples_split=2,
                        # min_samples_leaf=1,
                        # min_weight_fraction_leaf=0.0,
                        # max_features=None,
                        random_state=42,
                        # max_leaf_nodes=None,
                        # min_impurity_decrease=0.0,
                        # min_impurity_split=None,
                        # class_weight=None,
                        # presort=False,
                )
```

```
dt.fit(
    X=X_train,
    y=y_train,
    # sample_weight=None,
    # check_input=True,
    # X_idx_sorted=None
)
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier(max_depth=4, random_state=42)
```

```
dot_data = export_graphviz(
                        decision_tree = dt,
                        out_file=None,
                        # max_depth=None,
                        feature_names=X_test.columns,
                        class_names=['No detection', 'Detection'],
                        # label='all',
                        filled=True,
                        # leaves_parallel=False,
                        impurity=True,
                        # node_ids=False,
                        proportion=True,
                        rotate=True,
                        rounded=True,
                        # special_characters=False,
                        precision=4,
                        )
```

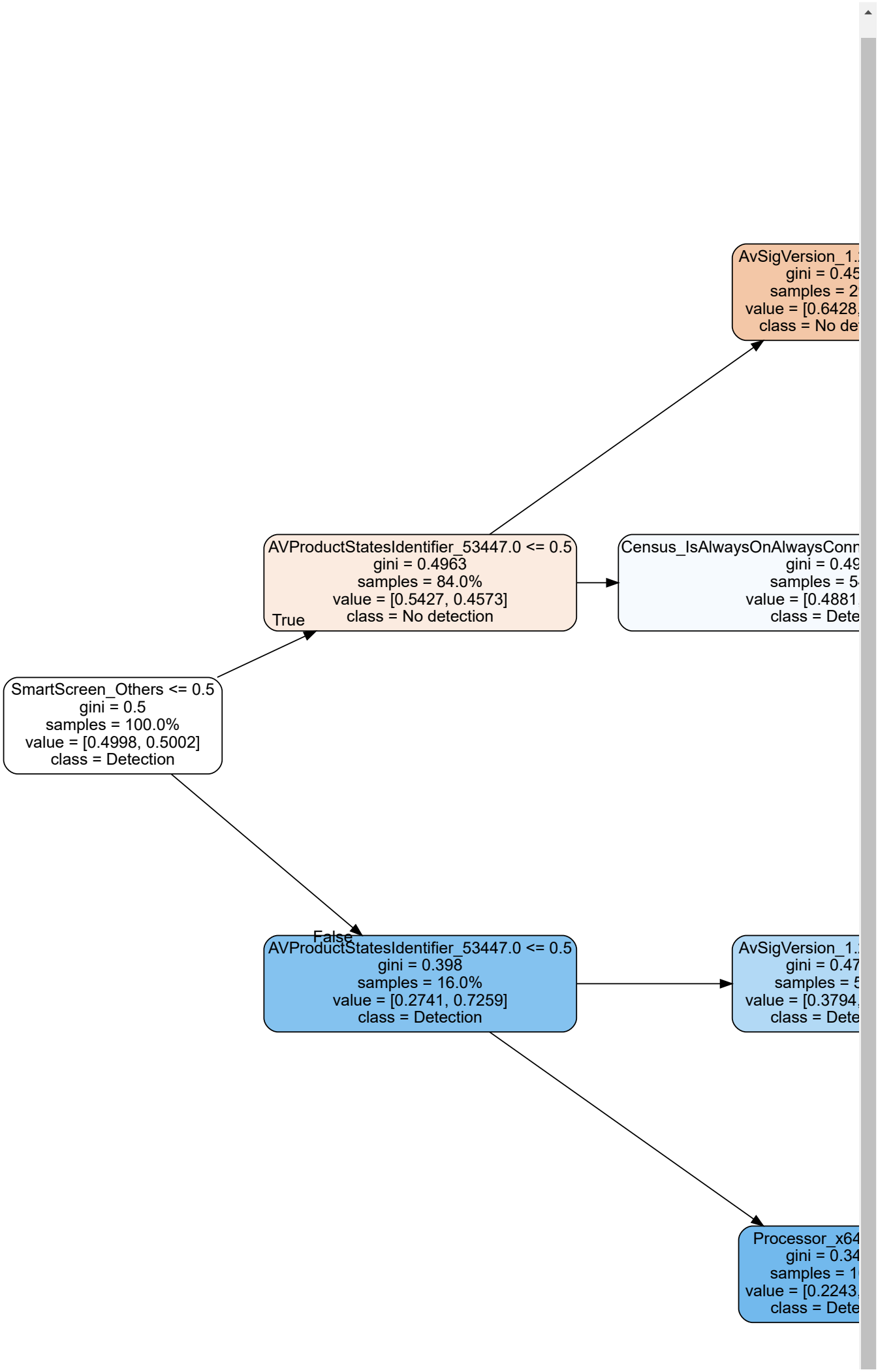
```
graph2 = pydotplus.graph_from_dot_data(dot_data)
```

```
graph2.write_png("tree.png")
```

```
True
```

3. Generamos el árbol de decisión

```
graphviz.Source(dot_data, format = 'png')
```


SmartScreen_Others <= 0.5
gini = 0.5
samples = 100.0%
value = [0.4998, 0.5002]
class = Detection
True
AVProductStatesIdentifier_53447.0 <= 0.5
gini = 0.4963
samples = 84.0%
value = [0.5427, 0.4573]
class = No detection
False
AVProductStatesIdentifier_53447.0 <= 0.5
gini = 0.398
samples = 16.0%
value = [0.2741, 0.7259]
class = Detection
Census_IsAlwaysOnAlwaysConn
gini = 0.49
samples = 5
value = [0.4881
class = Dete
AvSigVersion_1.
gini = 0.45
samples = 2
value = [0.6428
class = No de
AvSigVersion_1.
gini = 0.47
samples = 5
value = [0.3794
class = Dete
Processor_x64
gini = 0.34
samples = 1
value = [0.2243
class = Dete

```
graph = graphviz.Source(dot_data)
graph
```

4. Evaluating model

4.1 - Model results vs reality

```
y_test_pred = pd.DataFrame(dt.predict(X_test), index=y_test.index, columns=['DetectionPrediction'])

results_df = y_test.join(y_test_pred, how= 'inner')

results_df['Success'] = (results_df[TARGET] == results_df["DetectionPrediction"]).astype(int)

results_df['Success'].count()
```

```
104892
```

```
results_df['Success'].sum()
```

```
63231
```

```
results_df['Success'].mean()

# The accuracy of the model is 60%
# We observed an increase compared to the average of the initial target, which was 50%
```

```
0.6028200434732868
```

4.2 - Confusion matrix

```
confusion_matrix = pd.crosstab(results_df[TARGET], results_df['DetectionPrediction'])

confusion_matrix
```

HasDetections	DetectionPrediction 0	1
0	42292	10110
1	31551	20939

```
TP = confusion_matrix.iloc[1,1]
TN = confusion_matrix.iloc[0,0]
FP = confusion_matrix.iloc[0,1]
FN = confusion_matrix.iloc[1,0]

accuracy = (TP + TN) / (TP + TN + FP + FN)
accuracy

# We check that the precision is the same as in 4.1
```

```
0.6028200434732868
```

4.3 Metrics module

```
metrics.accuracy_score(results_df[TARGET], results_df['DetectionPrediction'])
```

```
0.6028200434732868
```

4.4 Using the model

```
dt.score(X_test, y_test)
```

```
0.6028200434732868
```

Validation strategies

```
for i in range(1, 20):
    dt = DecisionTreeClassifier(max_depth=i, random_state=42)
    dt.fit(X_train, y_train)
    train_accuracy = dt.score(X_train, y_train)
    test_accuracy = dt.score(X_test, y_test)
    print('Profundidad del árbol: {}. Train: {} - Test: {}'.format(i, train_accuracy, test_accuracy))
```

```
 Profundidad del árbol: 1. Train: 0.5720070113218957 - Test: 0.5747054112801739
 Profundidad del árbol: 2. Train: 0.5849346467985307 - Test: 0.5883766159478321
 Profundidad del árbol: 3. Train: 0.5965466379567471 - Test: 0.6001220302787629
 Profundidad del árbol: 4. Train: 0.5990022349610006 - Test: 0.6028200434732868
 Profundidad del árbol: 5. Train: 0.6017111547843284 - Test: 0.6042596194180682
 Profundidad del árbol: 6. Train: 0.6071575953944277 - Test: 0.609035960797773
 Profundidad del árbol: 7. Train: 0.6123221122220088 - Test: 0.6130400793196812
 Profundidad del árbol: 8. Train: 0.6193783784888068 - Test: 0.6180928955497083
 Profundidad del árbol: 9. Train: 0.6230393018096239 - Test: 0.6209339129771575
 Profundidad del árbol: 10. Train: 0.6267778563169314 - Test: 0.6198470808069252
 Profundidad del árbol: 11. Train: 0.6341037888104859 - Test: 0.6229550394691683
 Profundidad del árbol: 12. Train: 0.640682010402579 - Test: 0.6236605270182664
 Profundidad del árbol: 13. Train: 0.6474726962945409 - Test: 0.6231266445486786
 Profundidad del árbol: 14. Train: 0.6563389949621445 - Test: 0.6211817869808947
 Profundidad del árbol: 15. Train: 0.6666557710615452 - Test: 0.6186839797124661
 Profundidad del árbol: 16. Train: 0.6770787793108802 - Test: 0.61646264729436
 Profundidad del árbol: 17. Train: 0.6887806592113489 - Test: 0.616138504366396
 Profundidad del árbol: 18. Train: 0.7016510927610962 - Test: 0.6121248522289593
 Profundidad del árbol: 19. Train: 0.7141987440091196 - Test: 0.6097986500400412
```

Tree Pruning Method

```
for i in range(1, 20):
    dt = DecisionTreeClassifier(max_depth=i, random_state=42, min_samples_split=500)
    dt.fit(X_train, y_train)
    train_accuracy = dt.score(X_train, y_train)
    test_accuracy = dt.score(X_test, y_test)
    print('Profundidad del árbol: {}. Train: {} - Test: {}'.format(i, train_accuracy, test_accuracy))
```

```
 Profundidad del árbol: 1. Train: 0.5720070113218957 - Test: 0.5747054112801739
 Profundidad del árbol: 2. Train: 0.5849346467985307 - Test: 0.5883766159478321
 Profundidad del árbol: 3. Train: 0.5965466379567471 - Test: 0.6001220302787629
 Profundidad del árbol: 4. Train: 0.5990022349610006 - Test: 0.6028200434732868
 Profundidad del árbol: 5. Train: 0.6016580387093611 - Test: 0.6042310185714831
 Profundidad del árbol: 6. Train: 0.6070105047252877 - Test: 0.6090454944133012
 Profundidad del árbol: 7. Train: 0.611778693916575 - Test: 0.6132498188613049
 Profundidad del árbol: 8. Train: 0.6183487438048271 - Test: 0.6184551729397857
 Profundidad del árbol: 9. Train: 0.620587790657291 - Test: 0.6216298669107272
 Profundidad del árbol: 10. Train: 0.6221771870543867 - Test: 0.6219444762231628
 Profundidad del árbol: 11. Train: 0.6265081900901748 - Test: 0.6258437249742592
 Profundidad del árbol: 12. Train: 0.6290005597617131 - Test: 0.6277504480799299
 Profundidad del árbol: 13. Train: 0.6309576828316588 - Test: 0.6281794607787057
 Profundidad del árbol: 14. Train: 0.6334827393185616 - Test: 0.6283701330892728
 Profundidad del árbol: 15. Train: 0.6361385430669222 - Test: 0.6276455783091179
 Profundidad del árbol: 16. Train: 0.6375113893122285 - Test: 0.6271021622240018
 Profundidad del árbol: 17. Train: 0.6392315329707821 - Test: 0.6280936582389506
 Profundidad del árbol: 18. Train: 0.6409516766293356 - Test: 0.6274930404606643
 Profundidad del árbol: 19. Train: 0.6416340139000682 - Test: 0.6271784311482287
```

```
dt = DecisionTreeClassifier(max_depth=11, random_state=42, min_samples_split=500)
dt.fit(X_train, y_train)
```

```
DecisionTreeClassifier(max_depth=11, min_samples_split=500, random_state=42)
```

```
dt.score(X_train, y_train)
```

```
0.6265081900901748
```

```
dt.score(X_test, y_test)
```

```
0.6258437249742592
```

```
dt.score(val_df_virus8_X, val_df_virus8_y)
```

```
0.6216945996275606
```

F1

```
confusion_matrix
```

DetectionPrediction	0	1
HasDetections		
0	42292	10110
1	31551	20939

```
Precision = TP / (TP + FP)
Recall = TP / (TP + FN)
```

```
Precision
#When he says "yes", he is 67% correct
```

```
0.6743856484910947
```

```
Recall
# By having many false negatives, we see that the recall ends up being lower
```

```
0.3989140788721661
```

```
f1_score = 2 / ( 1/Precision + 1/Recall )
```

```
f1_score
```

```
0.5012987945749889
```

```
metrics.f1_score(y_test, y_test_pred)
```

```
0.5012987945749889
```

O of ROC curve

```
y_score = pd.DataFrame(dt.predict_proba(X_test)[:,1], index=y_test.index, columns=['DetectionScore'])
```

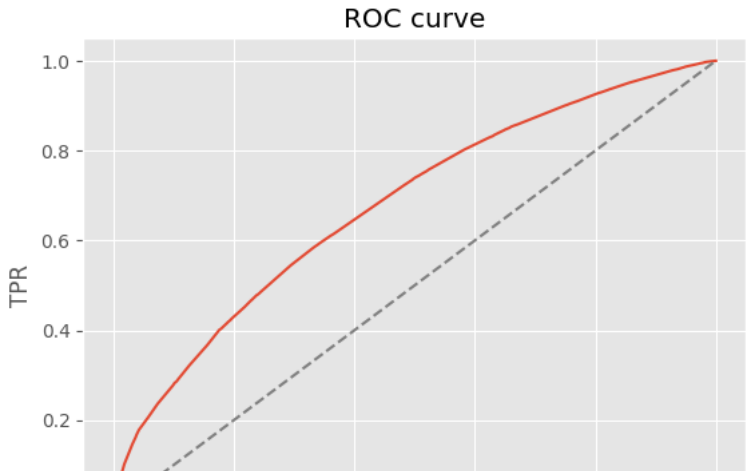
```
results_df = results_df.join(y_score)
```

```
print(metrics.roc_auc_score(results_df[TARGET], results_df['DetectionScore']))
```

```
0.677467230941152
```

```
fpr, tpr, _ = metrics.roc_curve(results_df[TARGET], results_df['DetectionScore'])
```

```
plt.clf()
plt.plot(fpr, tpr)
plt.plot([0, 1], [0, 1], color='gray', linestyle='--')
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC curve')
plt.show()
```



K-fold validation method

FPR

```
kf = model_selection.KFold(n_splits=10, random_state=42, shuffle=True)


scores_list = []
for train_index, test_index in kf.split(dev_df_virus8):
    #print("TRAIN:", train_index, "TEST:", test_index)
    X_train, X_test = dev_df_virus8_X.iloc[train_index], dev_df_virus8_X.iloc[test_index]
    y_train, y_test = dev_df_virus8_y.iloc[train_index], dev_df_virus8_y.iloc[test_index]
    dt = DecisionTreeClassifier(max_depth=8, random_state=42)
    dt.fit(X_train, y_train)
    _score = dt.score(X_test, y_test)
    scores_list.append(_score)
    #print(_scores)
scores_list
```

```
[0.6154902185104679,
 0.6196659421118865,
 0.6234126530145292,
 0.6182645006292187,
 0.6120867177668459,
 0.616548449834115,
 0.6168058574533806,
 0.6101418601990619,
 0.6134309575563437,
 0.6227154420387266]
```

```
np.mean(scores_list)
```

```
0.6168562599114575
```

```
np.std(scores_list)
```

```
0.0041075724005888
```

```
dt = DecisionTreeClassifier(max_depth=8, random_state=42)
```

```
dt.fit(X_train, np.ravel(y_train))
```

```
DecisionTreeClassifier(max_depth=8, random_state=42)
```

```
y_score = pd.DataFrame(dt.predict_proba(X_test)[:,1], index=y_test.index, columns=['DetectionScore'])
```

```
results_df = y_test.join(y_score)
```

```
print(metrics.roc_auc_score(results_df['HasDetections'], results_df['DetectionScore']))
```

```
0.670424877850003
```

Evaluation of alternative models

```
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
```

Random Forest

```
rf = RandomForestClassifier(n_estimators=3, max_depth=5, random_state=42)
```

```
rf.fit(
    X=X_train,
    y=np.ravel(y_train)
)
```

```
RandomForestClassifier
RandomForestClassifier(max_depth=5, n_estimators=3, random_state=42)
```

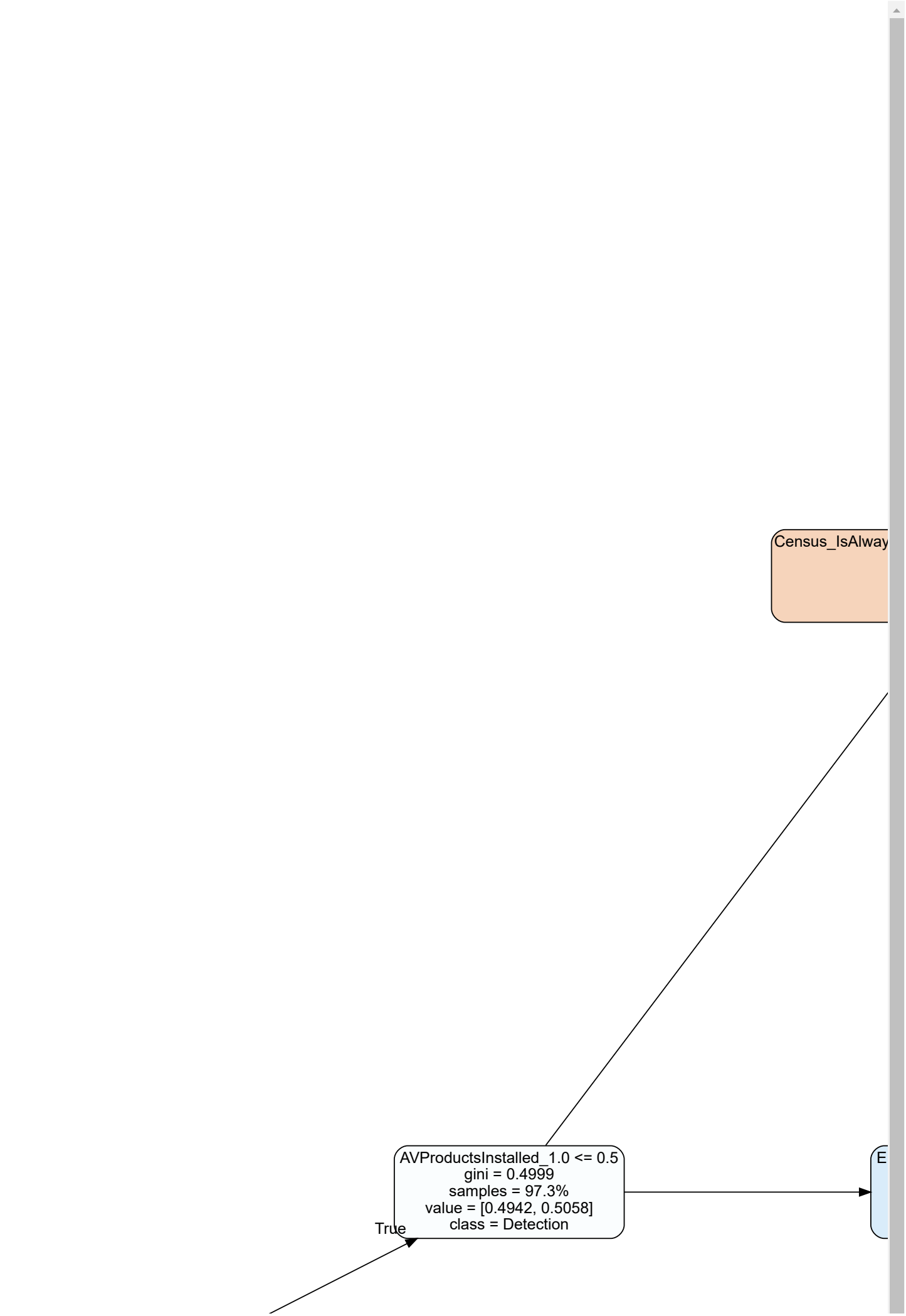
```
tree_list = rf.estimators_
```

```
tree_list
```

```
[DecisionTreeClassifier(max_depth=5, max_features='sqrt',
                        random_state=1608637542),
 DecisionTreeClassifier(max_depth=5, max_features='sqrt',
                        random_state=1273642419),
 DecisionTreeClassifier(max_depth=5, max_features='sqrt',
                        random_state=1935803228)]
```

```
plt.figure(figsize=(8,8))
dot_data = export_graphviz(
                        decision_tree = tree_list[0],
                        out_file=None,
                        feature_names=X_test.columns,
                        class_names=['No Detection', 'Detection'],
                        filled=True,
                        impurity=True,
                        proportion=True,
                        rotate=True,
                        rounded=True,
                        precision=4,
                        )

graph = graphviz.Source(dot_data)
graph
```


Census_IsAlway
AVProductsInstalled_1.0 <= 0.5
gini = 0.4999
samples = 97.3%
value = [0.4942, 0.5058]
class = Detection
True
E