Final Project Report- Windows Malware Attack Analysis

Srishti Saha (ss1078)

11 December, 2019

Summary

This report covers the analysis of malware attacks on Windows Machines. The objective is to identify hardware and software characteristics of a Windows machine that makes it susceptible to malware attacks. The report talks about characteristics like the RAM, system capacity, hardware featuures like touch screen, screen size etc. and checks if they are significant variables to predict or determine a malware attack. Software features like teh Operating System, presence and the characteristics of Anti Virus software, firewall etc. to determine if these specifications are linked to the possibility of a malware attack.

Introduction

Malware or malicious software is any form of software that is intentionally designed to cause damage to a computer, server, or a computer network. Enterprises like Microsoft take this to be a very serious problem. With over a billion enterprise and consumer customers, they want to develop data-science based techniques to predict if a machine will soon be hit with malware. Microsoft's next-generation detection solutions, like Windows Defender Antivirus uses data science, machine learning, automation, and behavioral analysis to provide 'real-time protection' to customers against threats. However, early detection of the possibility of a malware-attack will strengthen Microsoft's security. The goal of this project is to use data on different properties of a Windows machine to quantify the probability of it getting infected by any kind of malware.

This report will cover all steps from data gathering and processing steps. It will also talk about if gaming machines are more susceptible to malware attacks. Features like optical disk drive, pen and touch-activation are triggers for malware attack. Another objective is to find the best model that can accurately classify all malware detections based on all software and hardware characteristics. Following are the objective questions that we will answer: > * What are the most important factors that determine the odds of a malware attack on a Microsoft machine? * How does the machine being a gaming machine change the odds of a malware attack on a Windows machine? * Do features like pen, touch-activation and optical disk drive change the odds of a malware attack? * Does the presence and activation of AntiVirus products protect the machine from a malware attack?

Detailed code and notebooks are in the git repo: https://github.com/srishtis/MIDS_701_Final_Project

Data

The data is being sourced from Kaggle where the data had been put up by Microsoft as a part of a sponsored competition. The telemetry data containing data on the machines' properties and the machine infections was generated by combining heartbeat and threat reports collected by Microsoft's endpoint protection solution, Windows Defender. This data contains anonymized data from 16.8M devices and has over 8,000,000 records of data. The original dataset has 82 predictors. The **response variable 'HasDetections' is a binary variable**.

Data Dictionary

The original dataset had 83 features and a lot of them were not defined well or were imbalanced. There were also a lot of columns with missing values. The analysis took care of several of these issues and eliminated a lot of the features in the data preprocessing steps. The final dataset has 30 predictors. Of them 4 are numerical predictors and the rest are factor variables. A detailed data dictionary of these 30 predictors is mentioned in the appendix.

Data Preprocessing

Step 1: Check for missing values: In the original dataset with ~9 mill records, we checked for the proportion of missing values. 4 features had missing proportion greater than 60%. Since it would not make sense to impute these values and use them either, we removed them from the dataset

Table 1: Check for imbalance of data

Class	Proportion.of.records
0	50.02%
1	49.98%

Step 2: Check for imbalance in predictors: We checked for the distribution of values across the different levels of the predictors. We calculated the proportion of records across the most frequent value in a column versus the second most frequent value in the column. If this ratio was greater than 98%, we eliminated those features too.

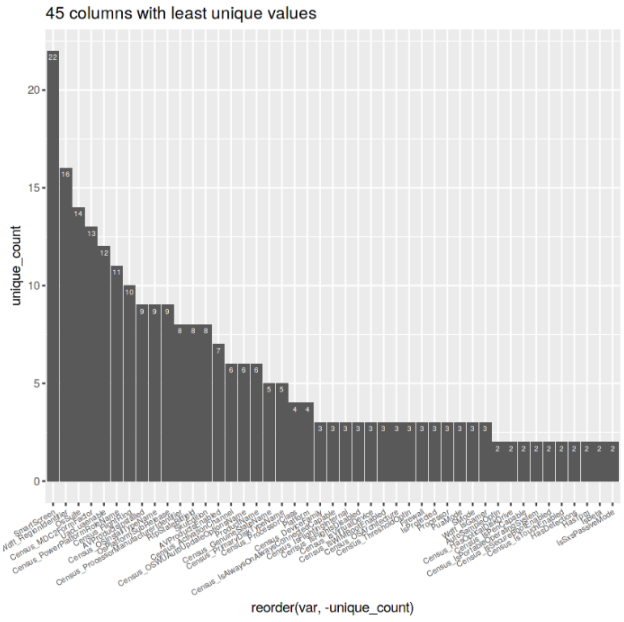

Figure 1: Top 45 columns with least unique values

Step 3: We eliminated all geographical features. 5 features were eliminated in this process

Step 4: Data Sampling: We first checked for imbalance of data across the 2 classes of the response variable. Since there was no imbalance of data, it was safe to assume that a random sample of rows from the original dataset would be effective. We sampled 75,000 rows from the actual dataset.

Step 5: We checked for correlation between the numerical variables. The following features were eventually selected to avoid multicollinearity: *"Census_SystemVolumeTotalCapacity", "Census_TotalPhysicalRAM", "Census_InternalPrimaryDiagonalDisplaySizeInInches", "Census_InternalBatteryNumberOfCharges"*

Step 6: Missing Value Imputation: We imputed the missing values of the columns. The categorical columns were imputed using a simple logic- All missing values were imputed with the string "Missing". This is because a lot of these features had missing and blank values implying the information was either not enabled or that setting was not activated on the machine. For the numerical variables, we tried imputing using 'pmm' from the 'mice' library. Due to lack of convergence, we then imputed the values using the arithmetic mean of the column.

Data Transformations

All necessary flags and character columns were converted into factors. Although we tried bucketing various variables into quartiles, none of these features were actually used in the analysis.

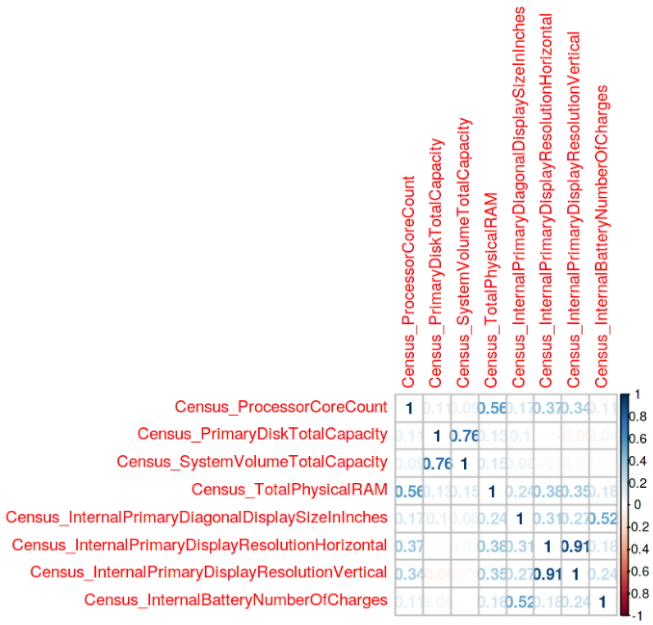

Figure 2: Correlation of numerical columns

EDA

We conducted chi-sqaured tests for various categorical variables to check if they are significant. The followig variables turned out to be highly significant: > * Wdft_IsGamer * Census_OSInstallTypeName * IsProtected * Census_IsTouchEnabled * Census_IsPenCapable * Platform

Interestingly, some of the features that did not seem relevant, based on the high p=value (above 0.05), are: HasTpm, Census_IsSecureBootEnabled.

The results of the chi-squared tests are in the Appendix.

For the numerical variables, we see that the system volume capacity does not really differ across the classes of 'HasDetections'. This shows that there is no strong relationship between the variable and the response. A similar trend was seen for display size (in inches).

However, for the number of Antivirus products installed show a clear higher number in machines without detections than in machines that had detections. All images are in the appendix.

Model

We built a logitic regression model with the following variables: ProductName+HasTpm+AVProductsEnabled+Census_TotalPhysi
+Census_InternalPrimaryDiagonalDisplaySizeInInches+Census_IsTouchEnabled+Census_IsPenCapable+Census_IsAlwaysOnA
+Wdft_IsGamer+Census_TotalPhysicalRAM:Census_IsTouchEnabled+Census_TotalPhysicalRAM:Wdft_IsGamer+Census_H

We included interaction terms between 'Wdft_IsGamer' and various hardware and software characteristics to check if those interactions have an effect on the model.

Model Assessment

The logistic regression model did not fare well according to the assumptions of linearit, normality and iid. The results can be clearly shown in the diagnostic plots (in the appnedix).

However, to quantify the results we can see that the most significant variable was 'Census_IsAlwaysOnAlwaysConnectedCapable' (level 2 i.e. when cable is always connected) as it had the highest z-score. The coefficient is -3.419e-01 i.e. when the cable is always connected, the odds of an attack decrease by ~30%. Other significant variables are touch enabled (level 2 i.e. touch is enabled) and the interaction term: Census_TotalPhysicalRAM:Census_IsTouchEnabled2

Other Models

We then created multiple other models. Owing to the large number of categorical variables, we adopted tree based methods. We tried random forest model, CART and bagging.

The most accurate model was the Bagging model with an accuracy of 60%

The AUC was 62.8%

```
          Reference
Prediction     0     1
         0 21934 14777
         1 15531 22758
```

Accuracy: 0.595893333333333

Sensitivity	0.606314106833622
Specificity	0.585453089550247

Figure 3: Bagging: Model Results

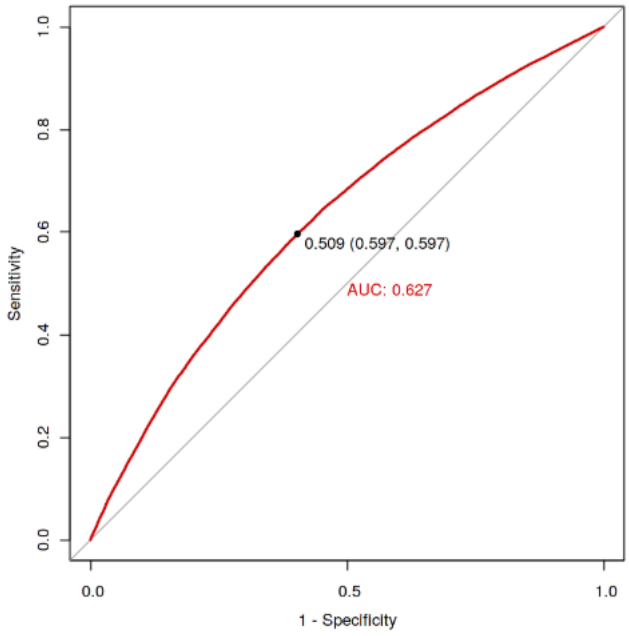

Figure 4: Bagging: ROC plot

Results

- What are the most important factors that determine the odds of a malware attack on a Microsoft machine? The top 3 most significant factors are: if the cable is always connected, is touch enabled and
- Does the machine being a gaming machine change the odds of a malware attack on a Windows machine? It is a significant variable and hence it changes the odds of a malware attack on a machine. Since the logistic regression model is not accurate according to the assumptions, the inference might not be accurate
- Do features like pen, touch-activation and optical disk drive change the odds of a malware attack? Yes, these features were significant and thus they are significant to determine if the machine is susceptible to malware attacks
- Does the presence and activation of AntiVirus products protect the machine from a malware attack? Yes, presence and activation of AV products decreases the odds of a malware attack.

Conclusion

The logistic regression model cannot be completely trusted for deterministic purposes here. Owing to the large number of categorical variables, it is better to go with a tre based model

The possible limitation of the tree based model is that its accuracy is very low for the problem statement here. We have only considered a small sample of the data (~75k rows) which might have led to loss of information. Moreover, determinintic and inferential proporties of a bagging model are not too high.

Appendix

Detailed Data Dictionary

The detailed definitions of the 30 predictors in the input dataset are mentioned here:

1. ProductName: Defender state information e.g. win8defender
2. AVProductsInstalled: Number of antivirus products installed
3. AVProductsEnabled: Number of Antivirus products enabled/activated
4. HasTpm: True if machine has tpm (Trusted Platform Module: hardware-based security services)
5. Platform: Calculates platform name (of OS related properties and processor property)
6. Processor: This is the process architecture of the installed operating system
7. OsVer: Version of current OS installed
8. OsSuite: Product suite mask for the current operating system.
9. 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.
10. 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.
11. 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
12. Firewall: This attribute is true (1) for Windows 8.1 and above if windows firewall is enabled, as reported by the service.
13. Census_ProcessorManufacturerIdentifier: ID of the manufacturer of the processor
14. Census_PrimaryDiskTypeName: Friendly name of Primary Disk Type - HDD or SSD
15. Census_SystemVolumeTotalCapacity: The size of the partition that the System volume is installed on in MB
16. Census_HasOpticalDiskDrive: True indicates that the machine has an optical disk drive (CD/DVD)
17. Census_TotalPhysicalRAM: Retrieves the physical RAM in MB
18. Census_InternalPrimaryDiagonalDisplaySizeInInches: Retrieves the physical diagonal length in inches of the primary display
19. Census_PowerPlatformRoleName: Indicates the OEM preferred power management profile. This value helps identify the basic form factor of the device
20. Census_InternalBatteryNumberOfCharges: Number of charges the internal battery has had
21. Census_OSArchitecture: Architecture on which the OS is based. Derived from OSVersionFull. Example - amd64
22. Census_OSInstallTypeName: Friendly description of what install was used on the machine i.e. clean
23. Census_OSWUAutoUpdateOptionsName: Friendly name of the WindowsUpdate auto-update settings on the machine.
24. Census_GenuineStateName: Friendly name of OSGenuineStateID. 0 = Genuine
25. 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.
26. Census_IsSecureBootEnabled: Indicates if Secure Boot mode is enabled.
27. Census_IsTouchEnabled: Is this a touch device ?
28. Census_IsPenCapable: Is the device capable of pen input ?
29. Census_IsAlwaysOnAlwaysConnectedCapable: Retreives information about whether the battery enables the device to be AlwaysOnAlwaysConnected .
30. Wdft_IsGamer: Indicates whether the device is a gamer device or not based on its hardware combination.
31. HasDetections: (response) Does the machine have malware detected? (1 if TRUE)

Chi-squared test results

```
        Pearson's Chi-squared test

data:  table(trainsample[, c("Census_OSInstallTypeName", "HasDetections")])
X-squared = 67.794, df = 8, p-value = 1.349e-11
```

```
        Pearson's Chi-squared test

data:  table(trainsample[, c("Wdft_IsGamer", "HasDetections")])
X-squared = 106.69, df = 2, p-value < 2.2e-16
```

```
        Pearson's Chi-squared test

data:  table(train[, c("IsProtected", "HasDetections")])
X-squared = 251.73, df = 2, p-value < 2.2e-16
```

```
        Pearson's Chi-squared test with Yates' continuity correction

data:  table(train[, c("Census_IsTouchEnabled", "HasDetections")])
X-squared = 173.75, df = 1, p-value < 2.2e-16
```

```
        Pearson's Chi-squared test

data:  table(train[, c("Platform", "HasDetections")])
X-squared = 16.985, df = 3, p-value = 0.0007117
```

```
        Pearson's Chi-squared test with Yates' continuity correction

data:  table(train[, c("HasTpm", "HasDetections")])
X-squared = 1.9056, df = 1, p-value = 0.1675
```

```
        Pearson's Chi-squared test with Yates' continuity correction

data:  table(train[, c("Census_IsSecureBootEnabled", "HasDetections")])
X-squared = 0.099298, df = 1, p-value = 0.7527
```

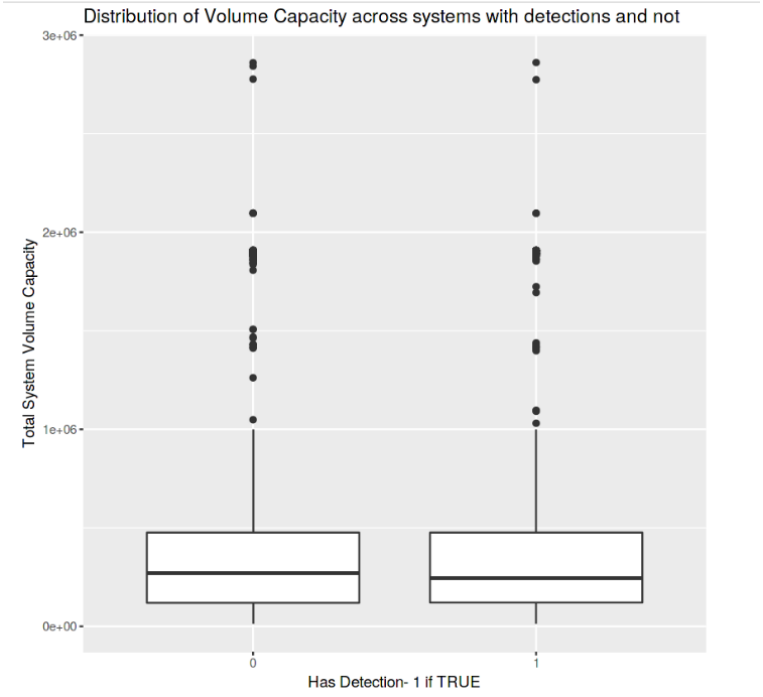

Figure 5: Volume Capacity

Box Plots for numerical variables

Logistic Regression

Diagnostic Plots:

Model Results:

The accuracy is 54% and AUC is 56.1 (for threshold of 50%)

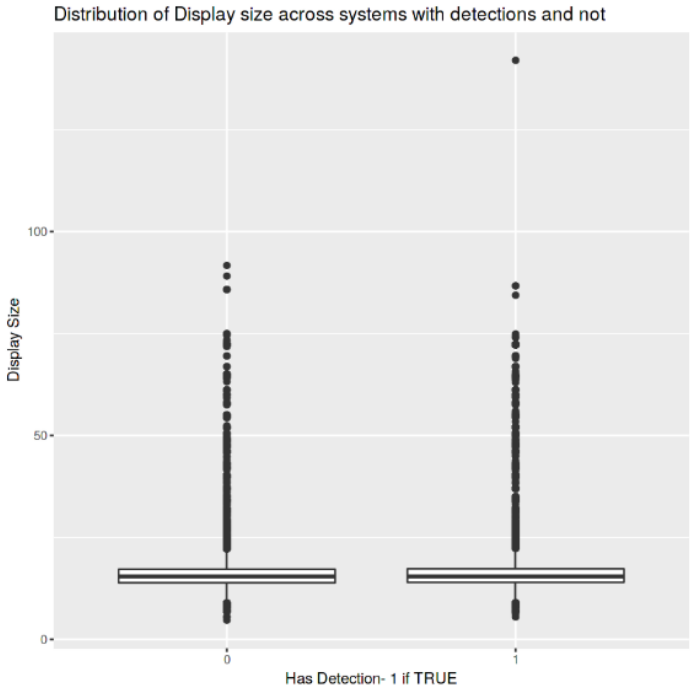

Figure 6: Display Size

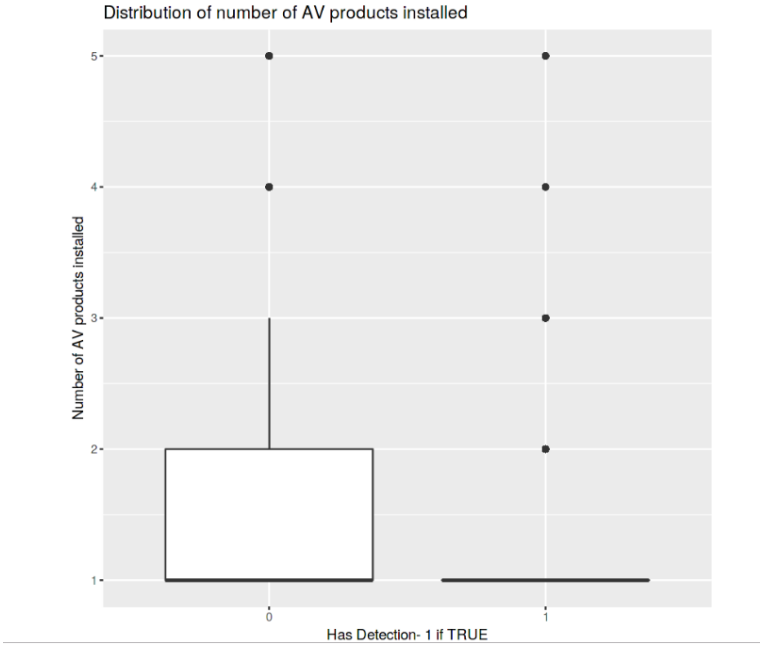

Figure 7: Nbr. of AV products

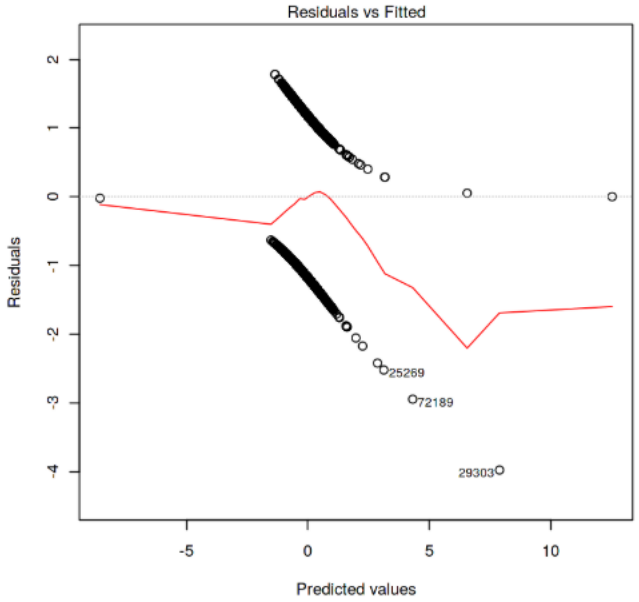

Figure 8: Residuals

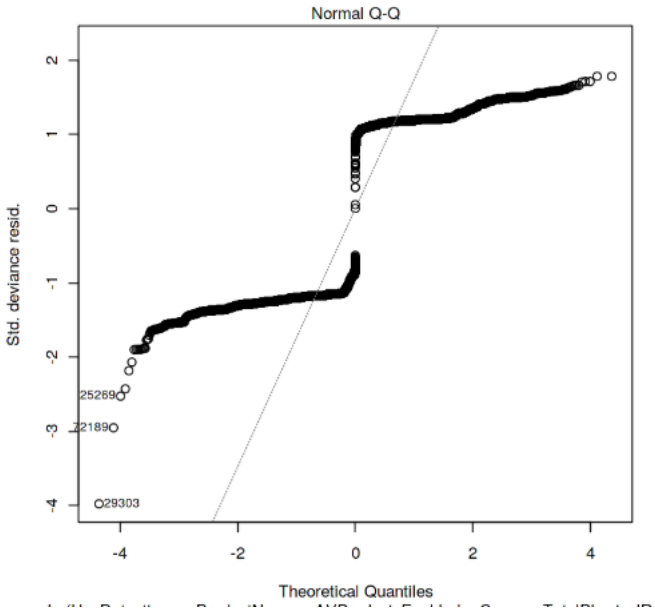

Figure 9: QQ plot

```
          Reference
Prediction     0     1
         0 20650 17676
         1 16815 19859
```

Accuracy: 0.54012

Sensitivity	0.529079525775942
Specificity	0.551181102362205

Figure 10: Logistic Regression: Model Results

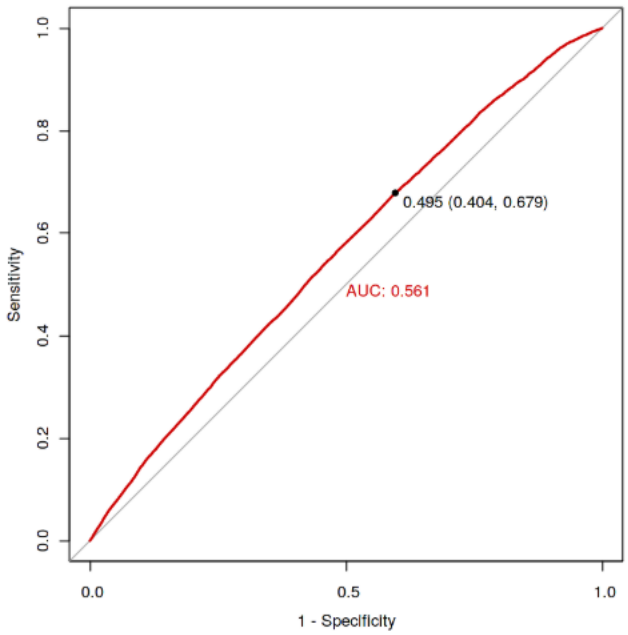

Figure 11: Logistic Regression: ROC plot