Methodology

Creating Dataset

To create dataset, 924 samples of benign data and 933 samples of malware were taken. To check whether the malicious files are as intended, the files were scanned using TotalVirus. TotalVirus is one of the most respected and widely used platform for malicious file detection. From the scanning it was found that, 933 files are malicious, and 924 files are non-malicious. Moving forward with results, these files were checked whether all of these files are Portable Executables (PE) or not. To do this, a bash script was written to automate the process. 1857 files were scanned from python PE file library, and it was confirmed that, the files are PE only. The dataset of malicious and non-malicious files was created by extracting the features of these files. Total 27 features were selected to be extracted. These selected features are highly targeted by the attackers to make files malicious.

Pre-processing dataset

Preprocessing of the dataset was performed to ensure that the data is in a suitable form for analysis. This included handling of missing values, checking for and correcting any invalid or inconsistent data, and normalization of the data to ensure that all features were on the same scale. Regarding the used dataset, missing values were replaced by imputing the mean of the respective column. Invalid data was replaced by imputing mean of the column as well. After this, the data was normalized. But, it was thought that, this may impact the accuracy of the models and therefore only rows containing NaN (Not a Number) values were removed. For training purpose 80% of data was used and 20% for testing purpose.

Statistical Approach

To understand statistical properties of the dataset, multiple methods were implemented. The main objective to implement these methods was to find which features are dominant in the process of classification of malware and benign. Also, it was aimed to find the correlation between the features of malware and benign. To do this, Spearman's Rank Correlation Coefficient and Pearson's Correlation was used. To find dominant features in classification process irrespective of machine learning model, Dimension Reduction Methods like Principal Component Analysis and Linear Discriminant Analysis are implemented.

Correlation Coefficient

A correlation coefficient is a measure of the strength and direction of the relationship between two any variables. There are several different types of correlation coefficients, including Pearson's correlation coefficient, Spearman's rank correlation coefficient, and Kendall's rank correlation coefficient. The purpose of implementing Correlation Coefficient on this study is to find the way features of malware and benign data are behaving with and against themselves. For this study purpose, only Pearson's Correlation and Spearman's Rank Correlation Coefficient has been implemented.

Pearson's r Correlation Coefficient

Pearson's r was the very first correlation method to be used [1]. Pearson's correlation coefficient is a measure of the linear correlation between two continuous variables. It is defined as the covariance of the two variables divided by the product of their standard deviations. The resulting coefficient, denoted by "r," takes on a value between -1 and 1. A value of 1 indicates a perfect positive linear relationship, meaning that as one variable increases, the other variable also increases at a fixed rate. A value of -1 indicates a perfect negative linear relationship, meaning that as one variable increases, the other variable decreases at a fixed rate. A value of 0 indicates no linear relationship between the variables.

$$r = \text{cov}(X, Y) / (\text{std}(X) * \text{std}(Y)) \quad (1)$$

where cov(X, Y) is the covariance between X and Y, and std(X) and std(Y) are the standard deviations of X and Y, respectively.

To calculate Pearson's correlation coefficient, you need to first calculate the covariance between the two variables and the standard deviations of each variable. Then, you can use these values to calculate the Pearson's correlation coefficient using the formula above.

Spearman's Rank Correlation Coefficient

Spearman's rank correlation coefficient is a non-parametric measure of the correlation between two variables. It is based on the ranks of the values rather than the actual values themselves, and it is calculated by comparing the ranks of the values in each variable and determining the degree to which they are correlated. The formula for Spearman's rank correlation coefficient (r) is:

$$r = 1 - (6 * \text{sum}(d^2) / (n^3 - n)) \quad (2)$$

where d is the difference between the ranks of the values in each variable, and n is the number of observations.

To calculate Spearman's rank correlation coefficient, you need to first rank the values in each variable from smallest to largest. Then, you can calculate the difference between the ranks of the values in each variable and sum the squares of these differences. Finally, you can use this sum to calculate the Spearman's rank correlation coefficient using the formula above. Spearman's rank correlation coefficient ranges from -1 to 1, with a value of 1 indicating a perfect positive correlation, a value of -1 indicating a perfect negative correlation, and a value of 0 indicating no correlation.

Inferential Statistics

Inferential statistics is a branch of statistics that deals with making predictions, inferences, or estimations about a population based on a sample. It allows to draw conclusions about a population based on observations made in a sample, and it helps to address the issue of sampling error, which is the difference between the characteristics of a sample and those of the population from which it was drawn.

Dimension Reduction Technique

Dimensionality reduction is a method for reducing the number of dimensions, or features, in a dataset by mapping the data onto a lower-dimensional subspace (where the number of dimensions (a), is always less than the original number of dimensions(b)). This can be useful in addressing the "curse of dimensionality," which refers to the challenges that arise when working with high-dimensional data. In this study, we applied two commonly used dimensionality reduction techniques, linear discriminant analysis (LDA) and principal component analysis (PCA), to datasets in the fields of machine learning and data mining.

Principal Component Analysis

Principal Component Analysis (PCA) is a statistical method that aims to represent a high-dimensional dataset with a set of linearly uncorrelated variables, known as principal components, while minimizing the loss of information. This technique allows for the reduction of the number of variables in a dataset, while preserving the maximal amount of variance. The first principal component is chosen such that it accounts for the largest proportion of variance in the data, and subsequent components are selected to account for the maximal remaining variance, in descending order. The benefits of PCA include decreased sensitivity to noise, reduction in computational and storage requirements, and increased efficiency in low-dimensional spaces.

Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a supervised method used for dimensionality reduction and feature extraction in pattern recognition and machine learning. It projects a dataset onto a lower-dimensional space, while maximizing the class separability. LDA assumes that the data within each class is normally distributed, and that the classes have equal covariance matrices. It aims to find a linear combination of features that maximizes the ratio of between-class variance to within-class variance, hence maximizing the separation between different classes and minimizing the variance within each class. It is a powerful technique for increasing class separability and is commonly applied in various applications such as pattern recognition, computer vision, and bioinformatics.

K-Nearest Neighbor

KNN is a machine learning algorithm which is used for classification and regression based operations. K-nearest neighbors (KNN) is a non-parametric, instance-based algorithm used for both classification and regression. It works by storing all available cases and classifying new cases based on a majority vote of its K nearest neighbors. The distance between the new data point and all the training data points is calculated and the K-nearest data points are selected. The new data point is assigned to the class to which the majority of the K-nearest data points belong. It is simple to implement but can be computationally expensive and sensitive to the choice of K and distance metric. KNN is used in scientific applications such as text classification and prediction of protein-ligand binding sites.

Random Forest

Random Forest is a machine learning algorithm that is used for both classification and regression. It is an ensemble method, which means that it combines multiple decision trees to improve the accuracy and robustness of the predictions. The algorithm works by first creating multiple decision trees, each of which is built from a bootstrap sample of the training data and a random subset of

the features. This process is known as random subspace method.

The final predictions are made by combining the predictions of all the decision trees. For classification, the majority vote of all the decision trees is taken and for regression the mean of all the decision tree's predictions is taken.

One of the main advantages of Random Forest is that it can handle large number of input variables and it can also identify important variables. In addition, it is less prone to overfitting compared to a single decision tree.

Support Vector Machine

Support Vector Machine (SVM) can be used for classification and regression. It is a powerful algorithm that can be used to solve complex problems in a variety of fields. SVM works by finding the best boundary, or hyperplane, that separates the data into different classes.

The algorithm finds the hyperplane that maximizes

the margin, which is the distance between the hyperplane and the closest data points from each class. These closest data points are known as support vectors and they define the boundary.

SVM also allows for non-linearly separable data by using kernel trick which maps the data into higher dimensions where it becomes linearly separable. This is done by transforming the input data into a higher dimensional space using a kernel function.

Naïve Bayes

Naive Bayes algorithm can be used as a classifier to identify and categorize different types of malware. The algorithm can be trained on a dataset of known malware samples and their corresponding features, such as file size, number of API calls, and presence of certain keywords in the code.

Once the algorithm is trained, it can be used to classify new, unseen samples as either malware or benign software based on their features. By assuming that the features of malware samples are independent from each other, the algorithm can quickly and efficiently determine the probability of a given sample being malware.

Naive Bayes algorithm can be particularly useful for detecting unknown or zero-day malware, which are not included in the training dataset. Since the algorithm is based on probability and assumes independence among features, it can still make predictions even when the number of features and their interactions are not fully understood.

Logistic Regression

Logistic Regression is a type of generalized linear model (GLM) that is used for classification problems. It is used to predict a binary outcome (1 / 0, Yes / No, True / False) given a set of independent variables.

The algorithm works by fitting a logistic function, also known as sigmoid function, to the data. The logistic function takes in a linear combination of the input features and outputs a probability value between 0 and 1. This probability value can then be used to predict the class of the data point.

The logistic regression algorithm finds the best coefficients for the input features by maximizing the likelihood of the observed outcomes, under the assumption that the data follows a logistic distribution. Once the model is trained, it can be used to predict the class of new data points by inputting them into the logistic function and interpreting the output probability value.

Logistic regression is widely used in many fields, such as medical research, social sciences, and marketing, where the goal is to predict a binary outcome. It's simple to implement and interpret, it's efficient and it can handle a large number of input variables.

Result

The correlation was established between malware and benign data of the same feature. This was done on every feature by diving it in two pieces of dataset. Two correlation methods have been implemented in this process which are Spearman's Rank Correlation Coefficient and Pearson's Correlation Coefficient. As the dataset contains discrete and continuous type of data, Pearson's Correlation Coefficient is used for continuous data and Spearman's Rank Correlation Coefficient is used for discrete data. It is observed that, nine features had positive correlation between both the file types. However, an inverse relationship was observed between the remaining features. Although the relationship was not statistically significant, yet, it was noticed that Size of initialized data, image base, Size of Stack reserve and Entropy may be the features of least interest while executing comparative analysis. Moreover, the magnitude of contribution of each feature in distinguishing type of file predominantly depends on the characteristics of the file in total. The

correlation among the features can be seen in the table 1. Here, correlation has been established in 20 features only. Remaining 6 features which are SizeOfHeapReserve, MinorOperatingSystemVersion, MajorOperatingSystemVersion, LoaderFlag, RVA and MinorImageVersion are not used because, there is no significant change in the values between benign and malicious files.

Feature	r
SizeOfInitializedData	-0.0223
ImageBase	-0.0191
DllCharacteristicsflags	-0.0134
e_lfnew	-0.0098
FileAlignment	-0.0087
SizeOfHeapCommit	-0.0076
SizeOfStackCommit	-0.0049
MinorSubsystemVersion	-0.0045
izeofCode	-0.0038
MajorSubsystemVersion	-0.0028
subsystem	-0.0005
SectionAlignment	0.0036
SizeofImage	0.0038
BaseofCode	0.0047
AddressofEntryPoint	0.0127
SizeOfHeaders	0.0181
MajorImageVersion	0.0196
Checksum	0.0300
SizeOfStackReserve	0.0303
Entropy	0.0335

Table 1 Correlation between the feature of malware and benign

malware and benign. The same methodology of implementing Spearman's Rank Correlation Coefficient and Pearson's Correlation Coefficient is implemented here. The output can be seen in figure 1.

Features		r
MajorSubsystemVersion	Subsystem	0.81
Checksum	Subsystem	0.75
Checksum	MajorSubsystemVersion	0.87
RVA	LoaderFlag	0.71
FileAlignment	SizeofHeader	0.78
SectionAlignment	BaseofCode	0.8
e_lfnew	SectionAlignment	-0.74

Table 2 Strong Correlations from Fig 1

From the table 2, it can be seen that, MajorSubsystemVersion and subsystem have a correlation of 0.81. Checksum and subsystem have a correlation of 0.75. Checksum and MajorSubsystemVersion have a correlation of 0.87. RVA and LoaderFlag have a correlation of 0.71. FileAlignment and SizeOfHeaders have a correlation of 0.78. SectionAlignment and BaseofCode have a correlation of 0.8. e_lfnew and SectionAlignment have a correlation of -0.74. To understand the correlation precisely, correlation between the features of only malware and benign is established. MajorSubsystemVersion and subsystem have a correlation of 0.8. Checksum and subsystem have a correlation of 0.83. Checksum and MajorSubsystemVersion have a correlation of 0.93. RVA and LoaderFlag have a correlation of 0.71. FileAlignment and SizeOfHeaders have a correlation of 0.73. On the other hand, the correlation among benign features

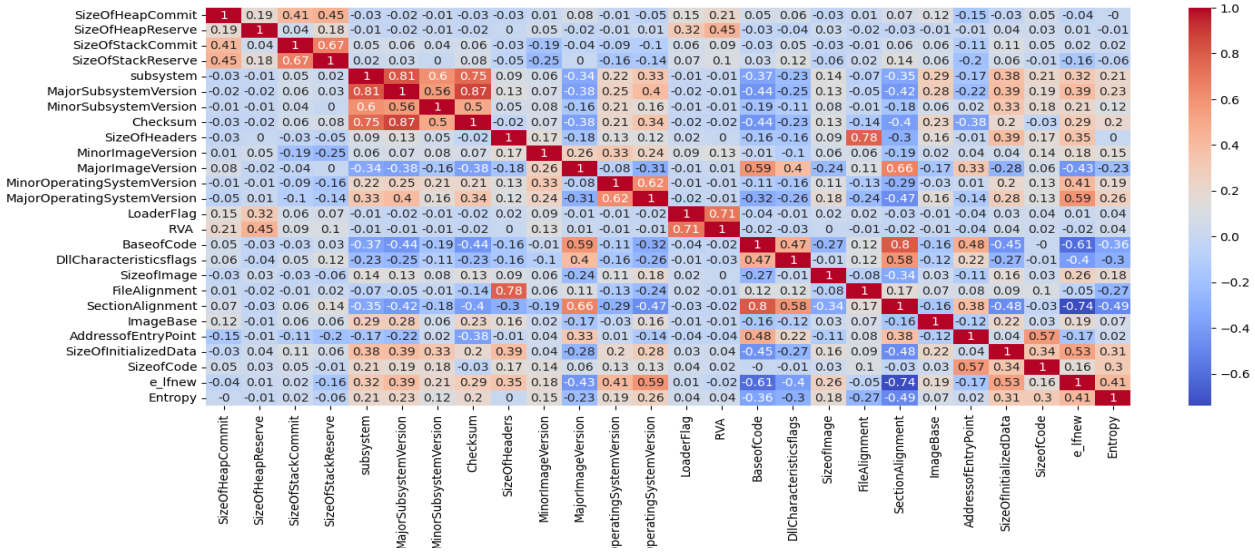

Figure 1 Correlation among the features.

After implementing this, the correlation is established between different features combining revealed that there is a strong correlation (1.0) between the features SizeOfHeapCommit and

SizeOfStackCommit, SizeOfHeapCommit and SizeOfStackReserve, SizeOfStackCommit and SizeOfHeapCommit, SizeOfStackCommit and SizeOfStackReserve, SizeOfStackReserve and SizeOfHeapCommit, and SizeOfStackReserve and SizeOfStackCommit. Additionally, there is a significant correlation 0.75 and 0.81 respectively between the features subsystem and MajorSubsystemVersion, subsystem and MinorSubsystemVersion, MajorSubsystemVersion and subsystem, and MinorSubsystemVersion and subsystem. Furthermore, the study also found that there is a strong correlation 0.99 between the features SizeOfHeaders and FileAlignment, BaseofCode and SectionAlignment, and FileAlignment and SizeOfHeaders, and SectionAlignment and BaseofCode. Also, there is a correlation 0.93 between the features AddressofEntryPoint and SizeofCode, and SizeofCode and AddressofEntryPoint. On the other hand, the study also revealed a negative correlation -0.99 between the features BaseofCode and e_lfnew, SectionAlignment and e_lfnew, e_lfnew and BaseofCode, and e_lfnew and SectionAlignment.

Features		**r**
MajorSubsystemVersion	subsystem	0.8
Checksum	subsystem	0.83
MajorSubsystemVersion	Checksum	0.93
RVA	LoaderFlag	0.71
FileAlignment	SizeofHeaders	0.73

Table 3 Correlation among Malware Features

Features		**r**
SizeOfHeapCommit	SizeOfStackCommit	1
SizeOfHeapCommit	SizeofStackReserve	1
SizeOfHeapCommit	SizeOfStackCommit	1
SizeoOfHeapReserve	SizeOfStackCommit	1
SizeOfStackReserve	SizeOfHeapCommit	1
SizeOfStackReserve	SizeOfStackCommit	1
Subsystem	MajorSubsystemVersion	0.75
Subsystem	MinorSubsystemVersion	0.81
SizeOfHeader	FileAlignment	0.99
BaseOfCode	SectionAlignment	0.99
FIleAlignment	SizeOfHeader	0.99
SectionAlignment	BaseOfCode	0.99
AddressOfEntryPoint	SizeOfCode	0.93
BaseOfCode	e_lfanew	-0.99
SectionAlignment	e_lfanew	-0.99

Table 4 Correlation among Benign Features

After implementing correlations, Principal Component Analysis is implemented on the dataset. It is found that, ImageBase, Checksum and LoaderFlag are making highest impact on the dataset. The variance given by these features are 0.63, 0.35 and 0.01. The features and variance can be seen in table 2. It also represents features ranking from most important to the least important.

id	**variance**	**feature name**
1	0.63442648	ImageBase
2	0.35246731	Checksum
3	0.01309195	LoaderFlag
4	5.91E-06	SizeofCode
5	5.59E-06	SizeofImage
6	7.67E-07	SizeOfInitializedData
7	7.35E-07	SizeOfStackReserve
8	6.14E-07	SizeOfHeapReserve
9	4.78E-07	SizeofCode
10	1.18E-07	SizeOfStackCommit
11	4.79E-08	BaseofCode
12	1.15E-09	DllCharacteristicsflags
13	9.99E-11	SizeOfHeaders
14	1.63E-11	SizeOfHeapCommit
15	1.22E-11	SizeOfHeapCommit
16	9.40E-12	SectionAlignment
17	8.68E-12	MinorSubsystemVersion
18	5.49E-14	MinorImageVersion
19	8.98E-15	e_lfnew
20	5.20E-16	MajorImageVersion
21	9.02E-18	,LoaderFlag
22	8.35E-18	Entropy
23	4.03E-18	MajorOperatingSystemVersion
24	1.26E-18	MinorOperatingSystemVersion
25	4.02E-19	MinorOperatingSystemVersion
26	1.36E-19	MajorSubsystemVersion

Table 5 PCA - Important features in descending order

At first, accuracy was measured between machine learning models with normal dataset, PCA and implementing LDA. Furthermore, the dominant features were removed from the dataset and accuracy is measured between new dataset and LDA. The results can be found in table 3.

	Original Dataset			Without Dominant Features	
Model	**Normal**	**PCA**	**LDA**	**Normal**	**LDA**
KNN	86.82	76.3	96.55	88.7	96.55
SVM	54.03	53.76	97.84	75.8	97.84
Random Forest	97.84	85.21	97.31	97.84	97.58
Naïve Bayes	12.36	3.76	87.36	63.17	87.63
Logistic Regression	68.27	50	97.04	83.06	96.5

Table 6 Accuracy

Conclusion

Correlation Among Entire Dataset

The correlation between the features of malware and benign on the entire dataset says that, The correlation coefficient between MajorSubsystemVersion and subsystem is 0.81, which is a strong positive correlation. It means that as the value of MajorSubsystemVersion increases, the value of subsystem also increases.

The correlation coefficient between Checksum and subsystem is 0.75, which is also a strong positive correlation. It means that as the value of Checksum increases, the value of subsystem also increases. The correlation coefficient between Checksum and MajorSubsystemVersion is 0.87, which is a very strong positive correlation. It means that as the value of Checksum increases, the value of MajorSubsystemVersion also increases.

The correlation coefficient between RVA and LoaderFlag is 0.71, which is a strong positive correlation. It means that as the value of RVA increases, the value of LoaderFlag also increases.

The correlation coefficient between FileAlignment and SizeOfHeaders is 0.78, which is a strong positive correlation. It means that as the value of FileAlignment increases, the value of SizeOfHeaders also increases.

The correlation coefficient between SectionAlignment and BaseofCode is 0.8, which is a strong positive correlation. It means that as the value of SectionAlignment increases, the value of BaseofCode also increases.

The correlation coefficient between e_lfnew and SectionAlignment is -0.74, which is a strong negative correlation. It means that as the value of e_lfnew increases, the value of SectionAlignment decreases.

These correlation coefficients suggest that there may be strong linear relationships between these pairs of features in the dataset, which could be useful in building a machine learning model to classify malware and benign samples.

Correlation among Malware Dataset

The correlation coefficient between MajorSubsystemVersion and subsystem is 0.8, which is a strong positive correlation. This suggests that as the value of MajorSubsystemVersion increases, the value of subsystem also increases. This may be because MajorSubsystemVersion and subsystem both relate to the operating system and library support of the malware, and so changes in one feature are likely to be accompanied by changes in the other feature.

The correlation coefficient between Checksum and subsystem is 0.83, which is also a strong positive correlation. This suggests that as the value of Checksum increases, the value of subsystem also increases. Checksum is This could be because the Checksum is a measure of the integrity of the malware file, and subsystem is a measure of the operating system and library support of the malware. As the integrity of the file increases, the operating system and library support of the malware also increases.

The correlation coefficient between Checksum and MajorSubsystemVersion is 0.93, which is a very strong positive correlation. This suggests that as the value of Checksum increases, the value of MajorSubsystemVersion also increases.

The correlation coefficient between RVA and LoaderFlag is 0.71, which is a strong positive correlation. This suggests that as the value of RVA increases, the value of LoaderFlag also increases. RVA (Relative Virtual Address) is a memory address that is relative to the base address of the module, and LoaderFlag is a flag that indicates how the image should be loaded into memory. As the RVA increases, the LoaderFlag also increases.

The correlation coefficient between FileAlignment and SizeOfHeaders is 0.73, which is a strong positive correlation. This suggests that as the value of FileAlignment increases, the value of SizeOfHeaders also increases. FileAlignment and SizeOfHeaders are both related to the structure of

the malware file, and changes in one feature are likely to be accompanied by changes in the other feature.

Correlation among Benign Dataset

The output shows that there is a strong positive correlation (1.0) between the features SizeOfHeapCommit and SizeOfStackCommit, SizeOfHeapCommit and SizeOfStackReserve, SizeOfStackCommit and SizeOfHeapCommit, SizeOfStackCommit and SizeOfSta ckReserve, SizeOfStackReserve and SizeOfHeapCommit, and SizeOfStackReserve and SizeOfStackCommit. This means that these features are highly correlated and that their values are likely to increase or decrease together. This high correlation can be explained by the fact that these features are all related to the memory allocation of a program and are likely to be set in a similar way, thus their values are expected to be related.

Additionally, there is a significant correlation 0.75 and 0.81 respectively between the features subsystem and MajorSubsystemVersion, subsystem and MinorSubsystemVersion, MajorSubsystemVersion and subsystem, and MinorSubsystemVersion and subsystem. This means that these features are also closely related and that their values are likely to increase or decrease together. This can be explained by the fact that the subsystem and Major/MinorSubsystemVersion are related to how the program is executed and how it interacts with the operating system. These features are likely to be set in a similar way, so their values are expected to be related.

Furthermore, the study also found that there is a strong positive correlation (0.99) between the features SizeOfHeaders and FileAlignment, BaseofCode and SectionAlignment, and FileAlignment and SizeOfHeaders, and SectionAlignment and BaseofCode. This means that these features are closely related and that their values are likely to increase.