

**DATA 1202 - Data Analysis Tools Analytics**

**Final Project: Data Analytics using Machine Learning**

**Professor:** Sk. Md. Mizanur Rahman, Ph.D.

**Group 4**

**Student Name and IDs**

Afshaan Shaikh - 100893464

Temitayo Ajayi - 100940989

Ibitomi Majolagbe - 100957240

Komal Verma – 100925046

Arti - 100942338

Contents

[Problem Statement 3](#_Toc174629355)

[Procedure 3](#_Toc174629356)

[Testing and Training the data 3](#_Toc174629357)

[Exploratory Data Analysis (EDA) 4](#_Toc174629358)

[Key Insights from EDA 4](#_Toc174629359)

[Feature distribution Histograms: 4](#_Toc174629360)

[Box plots: 5](#_Toc174629361)

[Correlation heatmap 6](#_Toc174629362)

[Pair Plot 7](#_Toc174629363)

[Machine Learning Classifiers 8](#_Toc174629364)

[Classifier 1: Logistic Regression 9](#_Toc174629365)

[Classifier 2: Decision Tree 9](#_Toc174629366)

[Classifier 3: Support Vector Machine (SVM) 10](#_Toc174629367)

[Overall interpretation: 10](#_Toc174629368)

[Classifier 4: Neural Network (NN) 11](#_Toc174629369)

[Classifier 5: K-Nearest Neighbors (KNN) 12](#_Toc174629370)

[Conclusion 13](#_Toc174629371)

# Problem Statement

How to leverage the power of machine learning to enhance cybersecurity measures by developing a helpful application capable of accurately classifying and identifying potential cyber threats in a cybersecurity field.

Procedure

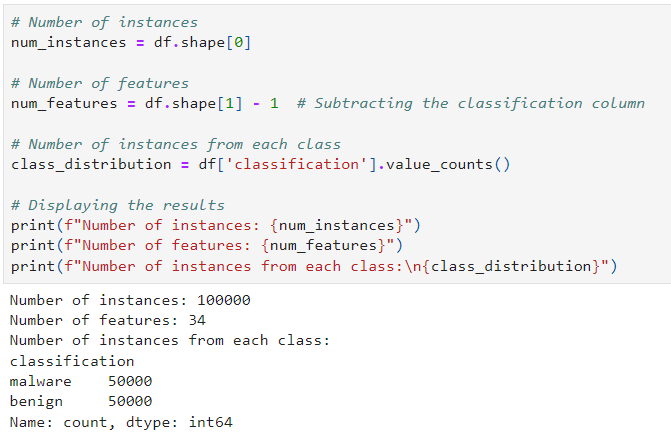
As instructed, the dataset was downloaded for further analysis, and it was read into pandas to extract this information:

Number of Instances: 100000 is the total number of instances in the dataset.

Number of Features: The dataset contains 34 features, which are used to predict whether an instance is Malware or Benign.

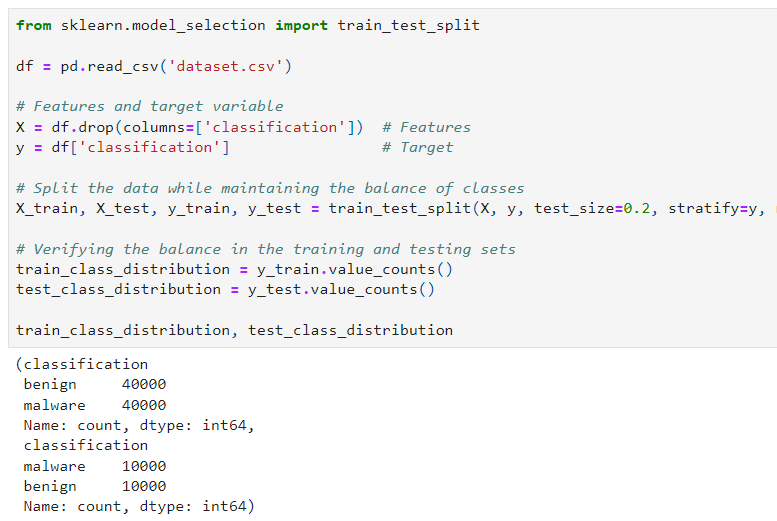
The number of instances of Malware is 50000

The number of instances from Benign is also 50000



# Testing and Training the data

To prepare the data for model training and testing, we split it into balanced training and testing sets. The dataset was loaded using **Pandas and Sklearn.** Then, it was divided into **80%** for training and **20%** for testing. **A stratified split** was performed to ensure that both training and testing datasets were balanced, with an equal representation of both classes (Malware and benign).



# Exploratory Data Analysis (EDA)

It’s time to explore the dataset for insights, patterns, or significant relationships between the features and the target variable. The first 5 rows were made visible by using this command **‘df.head()’**, the type of each column was identified as **‘df.dtypes’**, the dataset was queried to showcase the summary statistics (mean, median, standard deviation) to understand the central tendency and variability of the data **‘df.describe()’**, columns were checked if there were missing values **‘df.isnull().sum()’**.

Finally, plots such as histograms, box plots, pair plots and correlation heatmap (to identify any strong relationships between features) were generated to visualize the distribution of features and the relationship between features and class labels.

## Key Insights from EDA

### Feature distribution Histograms:

Many features have **highly skewed distributions.** For example, prio, total\_vm, exec\_vm, and nrcsw are heavily skewed to the right, suggesting that most values are concentrated towards the lower end of the range.

Features such as state, task\_size, and vm\_truncate\_count show signs of **bimodal or multimodal distributions**, indicating the presence of multiple distinct groups within the data.

The distribution of features like prio, total\_vm, and cached\_hole\_size suggests **the presence of** **potential outliers**, as most data points are concentrated in a narrow range with a few data points lying far away.

Features like free\_area\_cache and mm\_users have a **sparse distribution**, where a significant portion of the data points have zero or very low values.

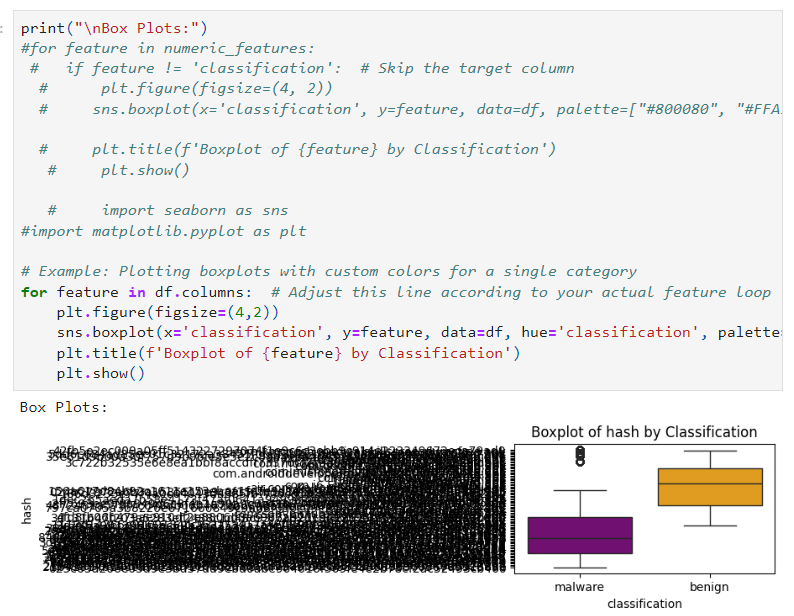


### Box plots:

The presence of **outliers** in the malware classification (indicated by points outside the whiskers of the box plot) suggests that some malware instances have hash values significantly different from the rest. These outliers could be important for further analysis or model development.

The **hash feature's median** (central line in the box) for Benign instances is higher than for Malware, indicating that Benign instances have higher hash values than Malware.

The **interquartile range (IQR),** represented by the height of the box, is wider for the benign classification than for Malware. This suggests that the hash values for benign instances are more variable.



### Correlation heatmap

**Strong Correlations:**

usage\_counter and prio: There seems to be a strong positive correlation between usage\_counter and prio, which may indicate that as the usage\_counter increases, the prio value tends to increase as well.

cached\_hole\_size and shared\_vm: These two features show a strong positive correlation, suggesting that when cached\_hole\_size increases, shared\_vm also tends to increase.

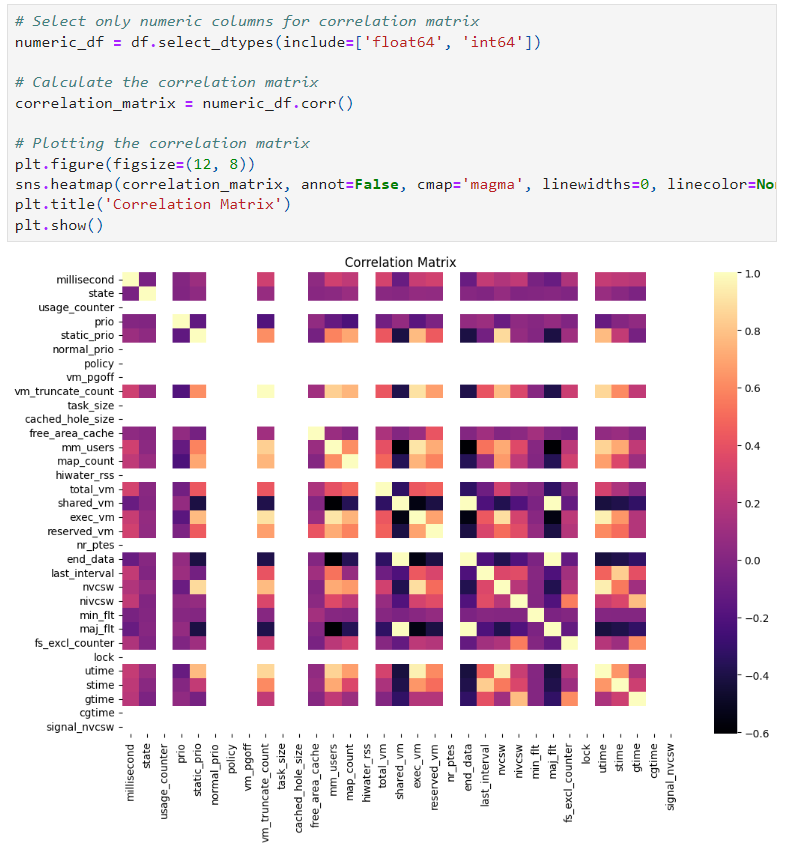
**Negative Correlations:**

mm\_users and total\_vm: There is a noticeable negative correlation between mm\_users and total\_vm. This might suggest that as the number of users (mm\_users) increases, the total virtual memory (total\_vm) decreases.

stimes and signal\_nvcw: These features have a moderate negative correlation, meaning the other tends to decrease as one increases.

**Weak or No Correlations:**

Several features show little to no correlation with others, indicating that they may be independent or have non-linear relationships. For example, milliseconds, fs\_excl\_counter, and cgtime have low correlation values with other features.



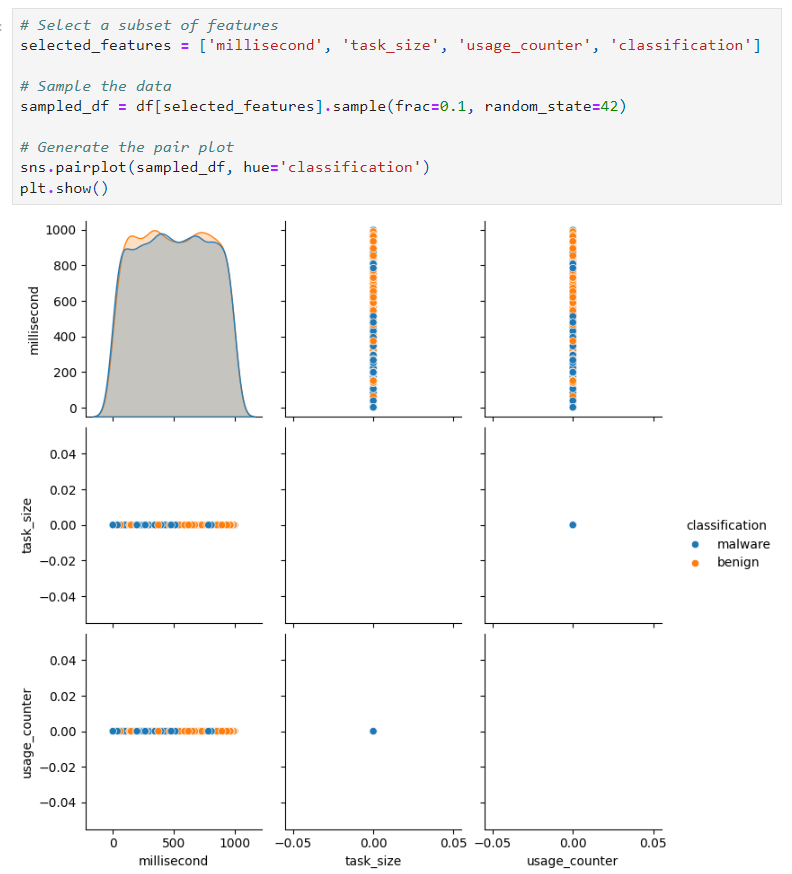
### Pair Plot

The features selected (millisecond, task\_size, usage\_counter) don't separate the Malware and Benign classifications.

The plot also suggests that the data might be highly imbalanced or have many overlapping feature values, which could impact the model's ability to learn effective boundaries.

The density plot on the diagonal indicates that the distribution of the millisecond feature appears fairly uniform.

The scatter plots between task\_size, usage\_counter, and millisecond don't show meaningful patterns or separations between the "malware" and "benign" classes.



# Machine Learning Classifiers

The idea is to implement and compare different machine-learning classifiers for detecting Malware. Each classifier was trained on the training dataset. We used the following classifiers:

Classifier 1: Logistic Regression

Classifier 2: Decision Tree

Classifier 3: Support Vector Machine (SVM)

Classifier 4: Neural Network (NN)

Classifier 5: K-Nearest Neighbors (KNN)

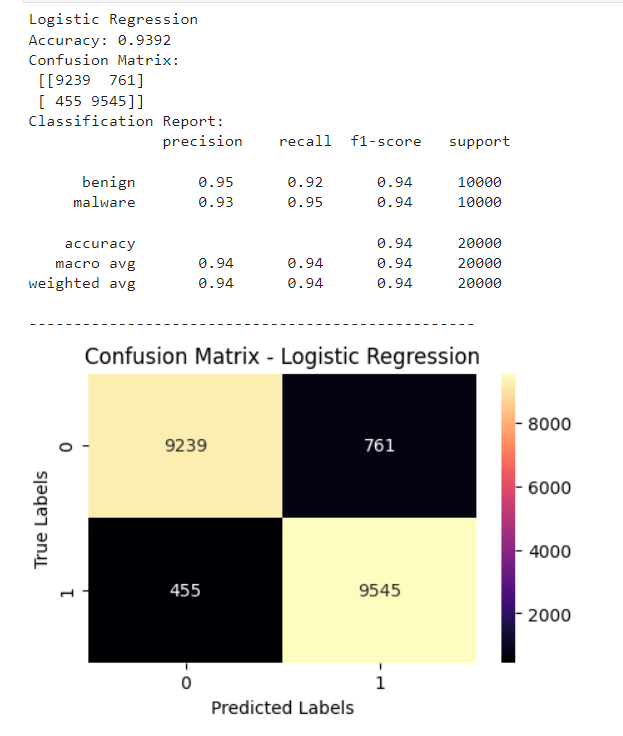
To evaluate the performance of each trained classifier, we used metrics like accuracy, precision, recall, F1-score, and confusion matrix.

Classifier 1: Logistic Regression

**Overall interpretation:**

The Logistic Regression model performs well, with a **high accuracy of 93.92%.** Both classes (Benign and Malware) are classified with similar precision, recall, and F1 Scores, indicating that the **model is balanced** in its predictions.

However, there are still some misclassifications**: 761** benign instances were falsely classified as Malware, and **455** malware instances were falsely classified as benign. Depending on the application, these misclassifications might be critical (e.g., in cybersecurity, false negatives could be more harmful).



## Classifier 2: Decision Tree

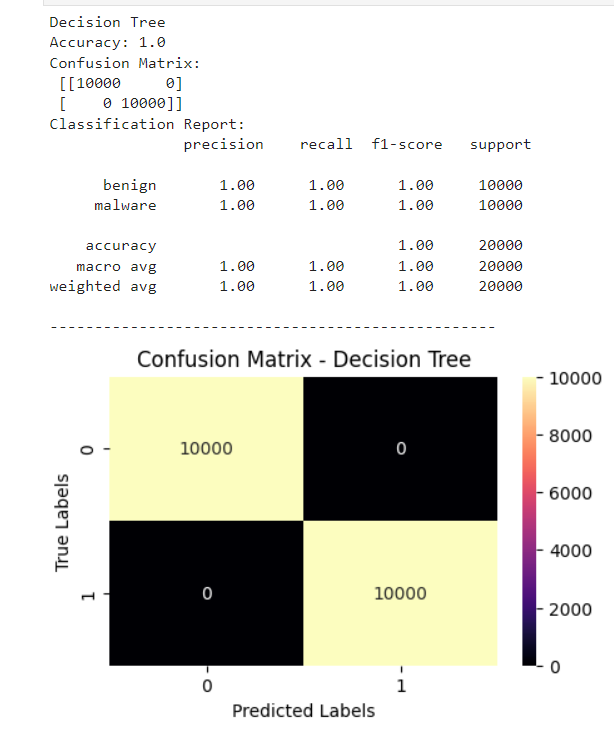
**Overall interpretation:**

The Decision Tree model perfectly classified all instances, achieving **100% accuracy, precision, recall, and F1 scores** for both classes.

However, a few considerations:

**Overfitting**: A perfect score on the test set often raises concerns about overfitting, especially with Decision Trees, which are prone to capturing noise in the training data. Overfitting occurs when the model performs exceptionally well on the training (and sometimes test) data but may need to generalize better to unseen data.

**Model Complexity:** The Decision Tree might have become too complex, leading to perfect classification on this dataset. This could indicate that the tree has memorized the data rather than learned general patterns.

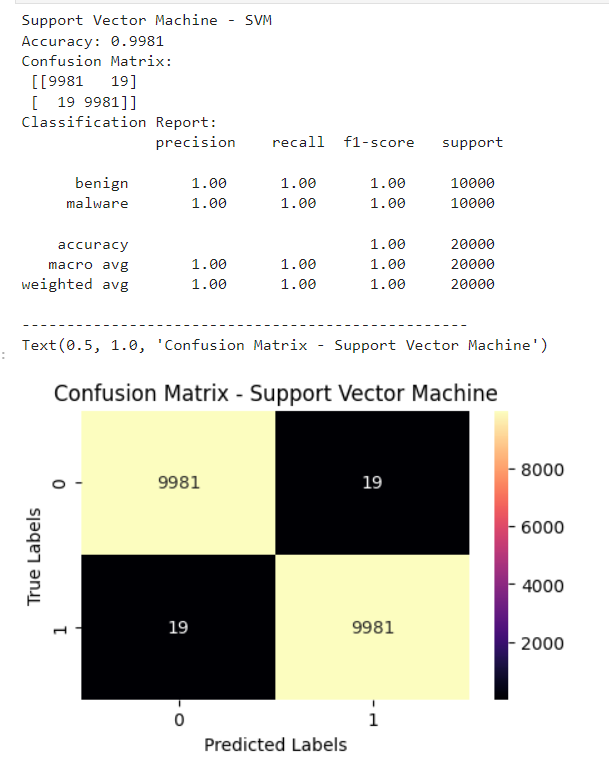


## Classifier 3: Support Vector Machine (SVM)

## Overall interpretation:

The SVM model performs exceptionally well, with an accuracy of **99.81%.** Both classes (Benign and Malware) have nearly perfect precision, recall, and F1 Scores, indicating that the model is highly effective at distinguishing between them.

**Misclassifications**: Only **19 instances** were misclassified in each class, which is a very small fraction considering the total number of instances (20,000). Further fine-tuning or investigation might be warranted if these misclassifications are critical (e.g., in security-related tasks). Given the high accuracy and low error rate, the model likely generalizes well to unseen data, although testing on a separate validation set would confirm this.

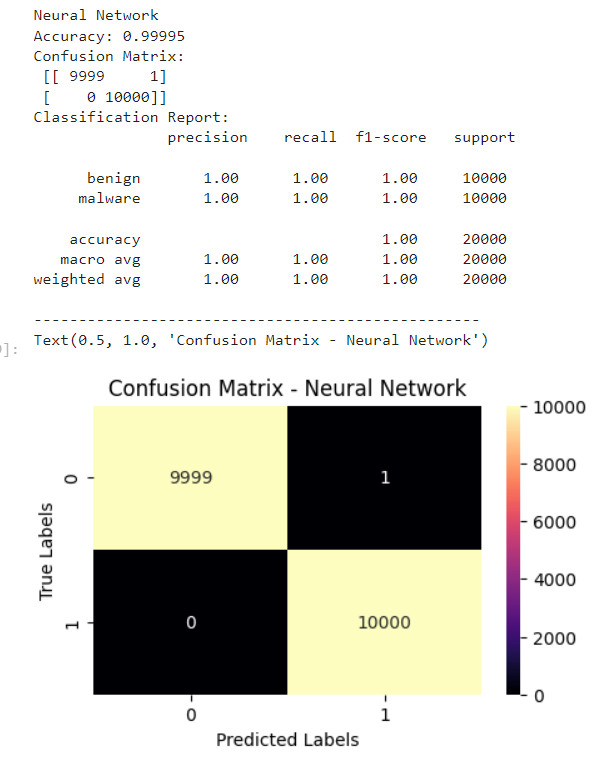


## Classifier 4: Neural Network (NN)

**Overall interpretation:**

The Neural Network model performs exceptionally well, with an almost perfect accuracy of **99.995%.** Precision, recall, and F1 scores for both the Benign and Malware classes are ideal, reflecting that the model is highly effective at distinguishing between the two classes.

**Misclassification:** Only 1 instance was incorrectly classified as Malware when it was actually benign, and no instances of Malware were misclassified as benign. While the error rate is extremely low, the single misclassification could be further investigated to understand whether it resulted from an edge case or noise in the data.

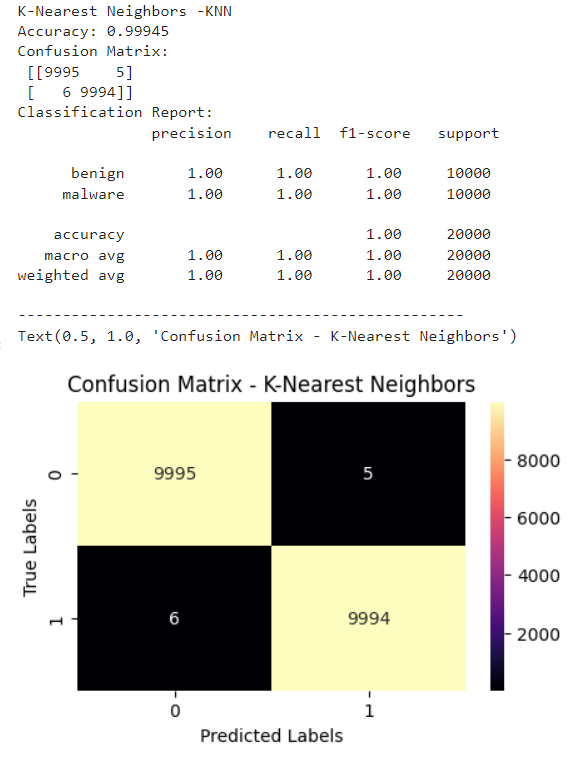


## Classifier 5: K-Nearest Neighbors (KNN)

**Overall interpretation:**

The KNN model demonstrates excellent performance with very few misclassifications, showing that it has learned to classify the data with high accuracy **(99.945%).** The misclassifications are minimal, with a total of 11 errors out of 20,000 instances, which may be acceptable depending on the application. 5 instances were incorrectly classified as Malware (false positives), and 6 instances were incorrectly classified as Benign (false negatives).

However, KNN can sometimes be sensitive to the distribution of the data and the choice of "k" (the number of neighbours), so further testing on a validation set might be helpful.



# Conclusion

Given the problem statement - “***How to leverage the power of machine learning to enhance cybersecurity measures by developing a useful application capable of accurately classifying and identifying potential cyber threats in a cybersecurity field***”, choosing the best model involves considering several factors beyond just accuracy. These factors include model interpretability, generalization capability, computational efficiency, and the ability to handle imbalanced data or unseen threats.

We recommend the Neural Network for the following reasons:

* It reports **a high percentage of accuracy**, which means it can reliably distinguish between Benign and Malware instances. This is crucial for cybersecurity, where false negatives (missed threats) can have severe consequences.

**Note:** **There was only 1 instance that was incorrectly classified as Malware.**

* Neural Networks excel at capturing complex, non-linear relationships in data, which is essential in cybersecurity. Cyber threats can evolve rapidly and present in diverse forms, requiring a model capable of adapting to these complexities.
* Neural Networks can be scaled and fine-tuned to handle large and diverse datasets. This is particularly beneficial in a cybersecurity application, where data continuously grows and new types of threats emerge. Neural networks can adapt to new threats, making them a powerful tool for enhancing cybersecurity measures.