**מטרות ויעדים:**

**מטרות:**

* **זיהוי אפקטיבי של נוזקות הנמצאות במחשב המשתמש**

**יעדים:**

* **מציאת סט נתונים רלוונטי לאימון המודל.**
* **בניית מודל למידת מכונה המבוסס על מאפייני סט נתונים.**
* **אינטגרציה של API המקבל כקלט את הקובץ הנבדק ומחזיר אינפורמציה רלוונטית בעבור שאילתה למודל למידת המכונה.**
* **חילוץ מאפיינים יחודיים באופן עצמאי מהקובץ הנבדק במידת הצורך.**
* **פיתוח בהתייחס במגבלות הטכנולוגיות שקיימות לנו.**

**מדדים:**

* **דיוק זיהוי מעל 70%. (Recall>0.7)**
* פחות מ10% זיהוי שגוי. (FPR<0.1)

**סקירה ספרותית :**

**נוזקות(-(Malware**

**נוזקות מחשב הן תוכנות זדוניות שנועדו לפגוע במחשבים או במערכות מחשב. הן יכולות לגרום לנזקים כספיים, לגנוב מידע או לפגוע בתפקוד של מחשבים.נוזקות יכולות להתפשט באמצעות דואר אלקטרוני, הורדות, התקנים ניידים או רשתות ציבוריות.כדי להגן על עצמכם מנוזקות, חשוב להשתמש בתוכנת אנטי-וירוס, להיות זהירים עם דואר אלקטרוני, להוריד תוכנות רק ממקורות מאובטחים, לשמור על מערכת ההפעלה מעודכנת ולהשתמש בחומת אש[1]..**

**טכניקות לזיהוי נוזקות:**

**זיהוי חתימה** היא השיטה העתיקה ביותר לזיהוי נוזקות, היא פועלת על ידי השוואת חתימת הקובץ של התוכנה לחתימה מוכרת של נוזקה. החתימה יכולה להיות מבוססת על קוד, על פונקציות מסוימות, או על התנהגות של התוכנה.**זיהוי חתימה** היא שיטה יעילה לזיהוי נוזקות מוכרות. עם זאת, היא אינה יעילה לזיהוי נוזקות חדשות או כאלו שעדיין לא נוצרה עבורן חתימה.

**כיצד פועל זיהוי חתימה?** תוכנות אנטי-וירוס משתמשות בבסיס נתונים של חתימות נוזקות כדי לזהות נוזקות. בסיס הנתונים הזה מתעדכן באופן קבוע על ידי היצרנים כדי לכלול חתימות של נוזקות חדשות.[1]

**זיהוי התנהגות** היא טכניקה המזהה תוכנות זדוניות על ידי ניתוח ההתנהגות שלהן. טכניקה זו יעילה לזיהוי נוזקות חדשות או כאלו שעדיין לא נוצרה עבורן חתימה.לדוגמה, תוכנת אנטי-וירוס יכולה לזהות נוזקה על ידי זיהוי התנהגות חריגה, כגון יצירת מספר רב של קבצים חדשים או יצירת קשר עם שרתים מרוחקים לא מוכרים.

**זיהוי מונחה ידע** הוא שילוב של זיהוי חתימה וזיהוי התנהגות. הוא משתמש בידע על נוזקות כדי לזהות אותן.

זיהוי מונחה ידע יכול להיעשות על ידי שימוש בטכניקות כגון:למידת מכונה, למידת חיזוק ולמידת עמיתים.

זיהוי מונחה ידע הוא שיטה יעילה לזיהוי נוזקות חדשות, מוכרות או כאלו שעדיין לא נוצרה עבורן חתימה.

**למידת מכונה בזיהוי נוזקות**

תוכנות אנטי-וירוס מבוססות חתימות קלאסיות מתקשות לעמוד בקצב הנוף המשתנה תמיד של איומי סייבר. תוכניות אלו מסתמכות על זיהוי דפוסי תוכנות זדוניות ידועים, דבר שחושף אותן להתקפות יום אפס – איומים חדשניים שטרם פוענחו. זה המקום שבו למידה מכונה (ML) נכנסת לתמונה, ומציעה גישה בעלת פוטנציאל לשנות את המשחק להגנה על אנטי-תוכנות זדוניות[2].

בעוד של-ML יש פוטנציאל עצום למהפכה בתוכנות אנטי-תוכנות זדוניות, עדיין ישנם אתגרים. דאגה אחת היא הפוטנציאל ל**תוצאות חיוביות כוזבות**, שבהן תוכנה לגיטימית מסומנת כזדונית. בנוסף, תוקפים יכולים לנסות "להערים" על מודלי ML על ידי הזנתם בנתונים מניפולטיביים.למרות האתגרים הללו, היתרונות הפוטנציאליים של אנטי-תוכנות זדוניות המופעלות על ידי ML חד משמעיים. ככל שהמחקר יתקדם והאלגוריתמים יהפכו מתוחכמים יותר, אנו יכולים לצפות לראות עתיד שבו מכונות תהיינה הקו הראשון שלנו להגנה מפני איומי סייבר מתפתחים ללא הרף.

**שימוש בFeatures בלמידת מכונה:**

זיהוי תוכנות זדוניות באמצעות למידת מכונה כרוך בניתוח מאפיינים שונים של קבצי הרצה על מנת לזהות דפוסים המעידים על התנהגות זדונית. בחירת המאפיינים תלויה במודל למידת המכונה הספציפית ובגישה שמערכת הזיהוי נוקטת. להלן כמה מאפיינים נפוצים המשמשים לזיהוי תוכנות זדוניות:

**רצפי בתים ו n-grams:** חילוץ רצפים זהים שחוזרים על עצמם, הן עבור בתים והן עבור מחרוזות.

**רצפי opcodes:** חילוץ רצפי opcodes שחוזרים על עצמם. ((operation codes

**קריאות API:** חילוץ קריאות API (ניתן זהות קריאות API שהן נפוצות אצל קבצים זדוניים)

**Metadata:** חילוץ אנפורמציה כגון גודל הקבוץ, תאריך יצירתו וחתימות דיגיטליות שלו.

**רמת שימוש במשאבים:** רמת השימוש במעבד או בזיכרון.

**זיהוי חיובי:** אחוז האנטי וירוסים שזיהו את הקובץ כזדוני.

**אנטרופיה:** מדידת רמת אקראיות של תוכן הקובץ. )קבצים זדוניים נוטים להכיל רמת אקראיות גבוה יותר עקב שיטות ערפול(

A diagram of a software development process

Description automatically generated**זיהוי שימוש תוכנות הצפנה/דיחסה/אריזה:** ניתן לחילוץ מכותרת הקובץ אם נעשה שימוש בכלי הצפנה, דחיסה ואריזה.

**מודלי קלסיפיקציה של למידת מכונה שהוכחו במחקרים כנותנים אחוז דיוק גבוהה בהקשר של זיהוי נוזקות:**

**Decision Trees C4.5(עצי החלטה) -** עץ החלטה הוא אלגוריתם למידת מכונה פופולרי המשמש גם כקלסיפיקציה וגם כרגרסיה. האלגוריתם פועל על ידי חלוקה רקורסיבית של הנתונים לקבוצות בהתבסס על ערכי המאפיינים הקלטיים ויוצר מבנה דמוי עץ שכל צומת פנימי מייצג החלטה בהתבסס על מאפיין וכל צומת עלה מתאים לניבוי סופי של התוצאה.

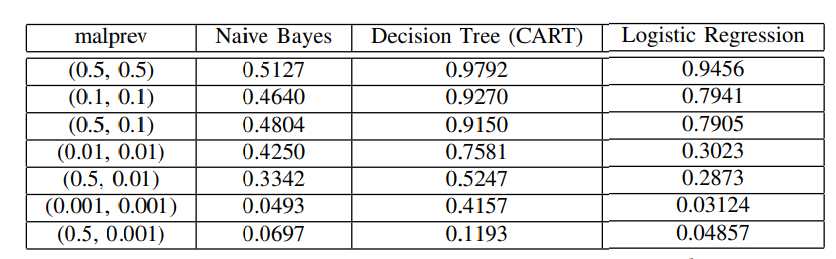
C4.5 הוא אלגוריתם שפותח על ידי רוס קינלן הוא בעצם יותר עץ החלטות מסויים שבוחר את המאפיין הטוב ביותר בחישוב רווח מידע של כל מאפיין.

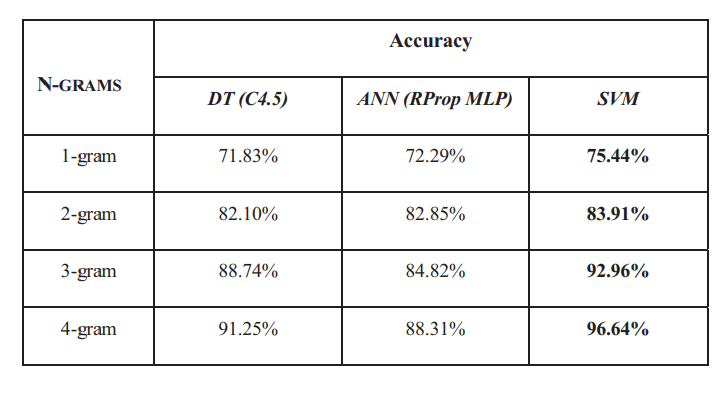
**Artificial neural networks(ANN)-Rprop MLP Algorithm –** רשתות נוירונים מורכבות ממספר רב של יחידות עיבוד פשוטות הנקראות נוירונים, אשר מחוברות באופן היררכי ומובנות בשכבות. השכבה הראשונה נועדה לקלוט מידע לרשת, השכבה האמצעית ידועה כשכבה החבויה (במודלים שונים עשויים להיות יותר מאחת כזו), ולבסוף השכבה האחרונה אשר נועדה להחזיר את המידע המעובד כפלט. אלגוריתם Rprop MLP זה אלגוריתם אימון אשר ידוע ביכולת המהירה שלו ויעילות שינוי המשקולות ברשת הנוירונים, האלגוריתם מתאים את קצב הלמידה לכל משקולת בנפרד מה שעוזר בעצם לפתור בעיות נפוצות בתהליך הלמידה.

**Support Vector Machine(SVM) –** אלגוריתם SVM הוא אלגוריתם למידה מונחת , האלגוריתם מוצע "מישור על" שבעצם יוצר

חוצץ בין מחלקות של מידע, האלגוריתם מתאים בעיקר למספר רב של ממדים ומתאים להחלטות שהגבולות לא ברורים.

בוצעו מספר נוסף של מודלים לקליספקציה לא נפרט את כולם כי הם אלגוריתמים בסיסיים בלמידת מכונה.

אחוז הההצלחה של כל האלגוריתמים לפי מציאת דפוס חוזר בקבצים.

****Malprev מדבר על אחוז זיהוי על אותה מערכת**.[2]**

**A graph of a positive rate

Description automatically generated**

KPI’s של המודלים שתוארו[3].

**מסקנות:**

בעקבות הסקירה הספרותית החלטנו להשתמש בשיטת הזיהוי לפי חתימה בשילוב עם למידת מכונה , המאפיינים שבחרנו להשתמש היא בהתאם למידע שהצלחנו לגשת עליו בחיפוש במאמרים שהם: אנטרופיה, זיהוי חיובי וMETA DATA(סוג קובץ,גודל ותאריך יצירה).

המודל יבצע מודל קלסיפקציה בינארית( קובץ נגוע או לא) עם אלגוריתם עץ החלטה על מנת שנוכל לפתח את המערכת בעתיד במידה ונצליח ליצור DATA SET עם יותר מאפיינים וככה נוכל להגדיל את האחוז דיוק בזיהוי כמה שיותר.

**סקר מתחרים:**

**:BitDefender**

חברת אבטחת מידע גלובלית מוקרת המספקת כלים רבים, בין היתר כלי נגד תוכנות ואתרים זדוניים בזמן אמת.

**יתרונות:**

* זיהוי בזמן אמת, במהלך שימוש בתוכנות או אתרים ייתכן והכלי יזהה התנהגות לא רואיה ויפעל בהתאם.
* מספקת חבילת אבטחה מקיפה הכוללת מגוון רחב של תכונות מעבר לאנטי-תוכנות זדוניות.
* חברה בעלת מוניטין רב, ידוע ביכולות גבוהות בזיהוי תוכנות נוזקה.

**חסרונות:**

* ניצול משאבים קבוע, מטבע הכלי לנתר את המערכת בכל רגע נתון, יש שימוש תמידי במשאבי המערכות המריצות אותו.
* יש מיגבלות משמעותיות בגירסה החינמית.
* יתכנו false-positives בזמן אמת עבור תוכנות ואתרים בטוחים.

**:Malwarebytes**

חברה המספקת כלי נגד תוכנות ואתרים זדוניים בצד המשתמש, נועדה לזהות ולהסיר תוכנות זדוניות מסוגים רבים ולמנוע כניסה לאתרים זדוניים, משתמשת בטכינקות מבוססות חתימה וגם מבוססות ניתוח התנהגותי.

**יתרונות:**

* חברה בעלת מוניטין רב, ידוע ביכולות גבוהות בזיהוי תוכנות נוזקה.
* כלי נוח ופשוט להתקנה על מחשב המשתמש, קל לשימוש עבור המשתמש הסטנדרטי.
* משלב שיטות זיהוי רבות, מבוסס חתימות, התנהגות באופן סטטי ודינמי, משלב למידת מכונה.
* זיהוי בזמן אמת, במהלך שימוש בתוכנות או אתרים ייתכן והכלי יזהה התנהגות לא רואיה ויפעל בהתאם.

**חסרונות:**

* ניצול משאבים קבוע, מטבע הכלי לנתר את המערכת בכל רגע נתון, יש שימוש תמידי במשאבי המערכות המריצות אותו.
* בגירסה החינמית חלק מהתכונות לא ניתנות לשימוש.
* יתכנו false-positives בזמן אמת עבור תוכנות ואתרים בטוחים.

**:Malwoverview**

כלי לניתוח תוכנות ואתרים זדוניים המנצל מספר ממשקים כגון: VirusTotal, Hybrid Analysis, Polyswarm.  
בנוסף, כלי זה מאפשר הורדת דוגמאות של תוכנות זדוניות דרך ממשקים נוספים, לייצר דוח מפורט לגבי הקובצים הנבדקים עוד..

**יתרונות:**

* משלב מספר רב של ממשקים שימושיים לזיהוי מדויק יותר.
* מאפשר ניתוח של קישורים.
* תורם למחקר אבטחת סייבר, ניתן להתנסות עם דוגמאות ולהסיק מסקנות לגבי הכלים בשימוש.
* הכלי חינמי וקוד פתוח.

**חסרונות:**

* פוטנציאל חשיפה לתוכן מזיק, הורדת דוגמאות של תוכנות יכולה לחשוף את המערכת של המשתמש לתוכנות זדוניות.
* מורכבות טכנית, עקב שילוב מספר רב של ממשקים מסוגים שונים, הכלי ניהיה מסורבל לשימוש.
* פוטנציאל לשימוש לא רואי, המידע וההיכולות המסופקים על ידי הכלי עלולים להיות מנוצלים לרעה, ויכול לסכן אחרים.
* לא נותן זיהוי בזמן אמת, רלוונטי לסריקות עצמאיות בלבד.
* למרות שהכלי חינמי, הוא משתמש בממשקים בעלי מגבלות משמעותיות בגרסה החינמית שלהם.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **תוכנה** | **Bitdefender** | **MalwareBytes** | **Malwoverview** | **התוכנה שלנו** |
| עלות | מוצר בסיסי חינם  הרחבת פיצ׳רים בתשלום | מוצר בסיסי חינם  הרחבת פיצ׳רים בתשלום | חינם | חינם |
| שימוש בלמידת מכונה | כן | כן | לא | כן |
| עיצוב | ממשק משתמש גרפי מלא | ממשק משתמש גרפי מלא | ללא ממשק(הרצה בטרמינל) | ממשק בסיסי |
| שיטת זיהוי נוזקות | זיהוי מונחה ידע | זיהוי מונחה ידע | כל טכניקות הזיהוי(מספר ממשקים) | מונחה ידע |
| זיהוי בזמן אמת | כן | כן | לא | לא |
| ניצול משאבים | בינוני | בינוני | נמוך | נמוך |
| מיגון מפני כתובות זדוניות | כן | כן | כן | לא |

**High Level Design**

**דרישות מערכת ((Software Requirements**

## דרישות מידע ופונקציונאליות:

* + המערכת תעמוד בדיוק של מינימום 70% לזיהוי נוזקות לא ידועות(שלא נוצרה להן חתימה).
  + המערכת תציג למשתמש האם הקובץ שהעלה הוא נוזקה שזוהתה בחתימה או חוזתה על ידי המודל למידת מכונה.
  + המערכת תאפשר למשתמש לעלות קובץ מכל סוג.
  + שימוש בAPI חיצוני שלVirusTotal לבדיקה ראשונית של הקובץ והוצאת חתימה

## דרישות לא פונקציונאליות:

דרישת עיצוב

* העיצוב יהיה פשוט ויאפשר העלאת קובץ וקבלת תוצאה באותו חלון

דרישת ביצועים

* מכיוון שמודל למידת מכונה צורך הרבה משאבים ממערכת, המערכת עלולה ליהיות עמוסה ואיטית, לכן יש צורך להפריד את המודל למידת מכונה מבקשות המשתמשים.
* דרישה שהמודל למידת מכונה יחושב בשרת אחר מהשרת בקשות משתמש
* יש הגבלה ל4 קריאות בדיקה בדקה ו150 בדיקות ליום מצד הAPI
* זמן החזרת תשובה לא יעלה על 2 שניות.

דרישת תאימות

* המערכת תיהיה זמינה בכל הפלטפורמות, לכן נגדיר ממשק WEB שיהיה זמין בכל פלטרפורמה ויהיה גמיש.

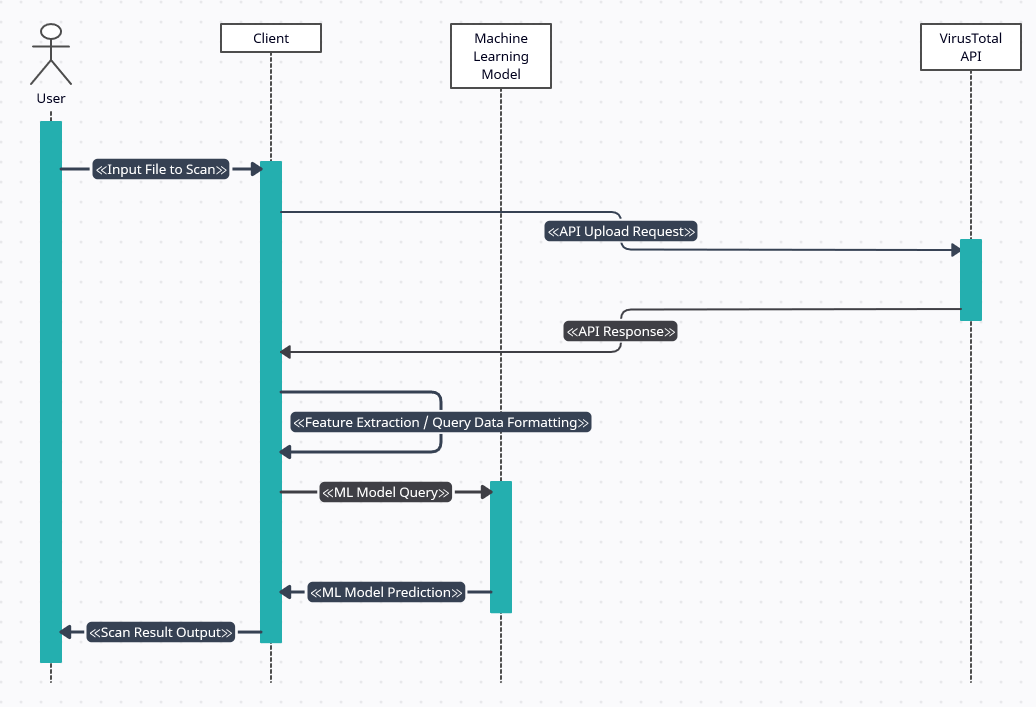
**השוואת חלופות:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Features** | **Advantages** | **Considerations** | **Algorithm** |
| - Suitable for both numerical and categorical data - Effective at capturing non-linear relationships between features - Can identify complex patterns in malware behavior | - Versatile and can handle various types of features - Effective for capturing complex relationships in data | - Prone to overfitting with complex trees - Sensitive to small variations in the data | Decision Trees |
| - Handles high-dimensional data well - Effective at separating different classes of malware - Can capture complex relationships in the data | - High accuracy in classifying malware samples - Works well when there is a clear margin of separation between classes | - Computationally intensive for large datasets - Selection of the kernel function is crucial for performance | SVM |
| - Simple and efficient, suitable for large datasets - Performs well in practice for malware classification tasks - Assumes conditional independence of features, which is beneficial for certain types of features | - Efficient for training and classification tasks - Handles large feature spaces well | - Conditional independence assumption may not hold true in all real-world scenarios | Naïve Bayes |
| - Optimization algorithm used in training machine learning models - Essential for training models such as logistic regression and neural networks, which are used in malware detection | - Widely used for training machine learning models - Can handle large-scale optimization problems | - Selection of learning rate is critical for convergence and performance - May get stuck in local minima due to the non-convex nature of the loss function | Gradient Descent |
| **Features** | **Advantages** | **Considerations** | **Algorithm** |

| **Service** | **Features** | **Accuracy** | **Ease of Use** | **Price** |
| --- | --- | --- | --- | --- |
| **PolySwarm** | - Crowdsourced threat detection marketplace - Supports Python 3.5+ - Metadata search, artifact scope attributes, sandboxing capabilities | Inferred from crowdsourced model and community engagement | Python interface for API interaction | Crowdsourced marketplace model, specific pricing not mentioned |
| **Malpedia** | - Comprehensive database of malware families, actors, samples - REST API for metadata, YARA rules, sample information - API calls for detailed analysis | Inferred from curated database and community engagement | REST API with token authentication | Free service offered by Fraunhofer FKIE |
| **Hybrid Analysis** | - Automated malware analysis service - API for report data, advanced search queries - Various authentication levels, community-contributed connectors | Inferred from sandboxing technology and community engagement | Comprehensive API documentation, OpenAPI for integration, community-contributed connectors | Free service, API keys for non-vetted accounts for database search |
| **ThreatFox** | - Focus on collecting and sharing indicators of compromise (IOCs) - Part of abuse.ch project | Inferred from focus on IOCs and community engagement | Affiliation with abuse.ch suggests community focus and accessibility | Likely free access or contribution-based participation, specific pricing not mentioned |
| **VirusTotal** | - Extensive database of file hashes, URLs, and IP addresses - RESTful API for querying file reports, URL scans, IP address reports - Code examples in various programming languages | Inferred from the variety of features and community engagement | Extensive documentation and code examples, RESTful architecture | Public API: Free with limitations. Premium API: Custom pricing based on specific needs |

**Design Tools, Analysis & Visualization**

**Sequence Diagram**

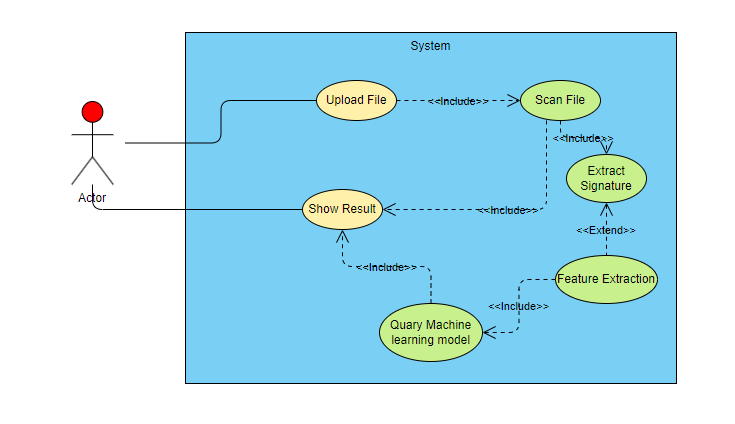
****

**Flow Diagram**

A diagram of a flowchart

Description automatically generated

**Use Case Diagram**



|  |  |  |
| --- | --- | --- |
| סיכון | רמת סבירות(1-5) | רמת חומרה(1-5) |
| חוסר נגישות לסט נתונים רלוונטי | 2 | 5 |
| פערי ידע של צוות המפתחים | 3 | 3 |
| אי זמינות API חיצוני בבדיקות | 1 | 2 |
| הידבקות במהלך בדיקות | 2 | 5 |

**ניהול סיכונים**

**חוסר נגישות לסט נתונים רלוונטי:**

ניתן לנסות ליצור קשר עם חברות או גופים המחזיקים בסט נתונים רלוונטי ולבקש גישה או גישה חלקית בשביל הפרויקט.

**רמת הסבירות:**

נמוכה(2), כיום יש הרבה מאוד מקורות נתונים חינמים כגון Kaggle, מאגרים אקדמיים חינמים, ונתונים חינמים מארגוני אבטחת סייבר.

**רמת החומרה:**

גבוה(5), אי יכולת השגת סט נתונים הינו קריטי עקב מבנה הפרוקיט המבוסס על למידת מכונה, ואין באפשרותנו ליצור סט נתונים מהסוג הזה בצורה פשוטה.

**פערי ידע של הצוות:**

נוכל להיעזר בקודים ממקורות פתוחים במידת הצורך, ו/או ליצור קשר עם בעלי ניסיון בתחום.

**רמת הסבירות:**

בינונית(3), בהתאם לציפיות שהצבנו עבור הפרוקיט שלנו, בחרנו בקפידה ככל שאפשר לממש מיבנים שברים לביצוע בהתאם לרמת הידע שלנו.

**רמת החומרה:**

בינונית(3), אי יכולת ביצוע מימוש המבינים עלולה להוביל לכך שהפרויקט לא יעמוד ביעדים שהוצבו עבורו.

**אי זמינות API חיצוני בבדיקות:**

ניתן יהיה לחפש API אלטרנטיבי, אשר יבצע חילוץ המפאיינים באופן דומה, ובמידת הצורך לחלץ באופן ידני.

**רמת הסבירות:**

נמוכה מאוד(1), על פי בדיקות שערכנו, זמינות האתר VirusTotal גבוה מאוד.

**רמת החומרה:**

נמוכה(2), אי זמינות האתר בדרך כלל נפתרת תוך זמן קצר, על כן אינו מהווה בעיה חמורה.

**הידבקות במהלך בדיקות:**

נידבוק בעקרונות בטיחות על מנת למינוע רשלנות שתגרום להדבקות במהלך בדיקות, למשל הסרת סיומות של הקבצים הנבדקים, מניעת הרצה של קבצים לא מזוהים ועוד.

**רמת הסבירות:**

נמוכה(2), הסבירות להדבקות כאשר נוקטים בעקרונות בטיחות נמוכה יחסית, לא אמור להתאפשר מצב של הרצת קובץ זדוני על מחשבי המפתחים.

**רמת החומרה:**

גבוה מאוד(5), הידבקות יכולה לחשוף את פרטיות הפרויקט לגורמים זדוניים, תיתכן התעכבות ואף מחיקה של רכיבים חיוניים לפרוקיט.

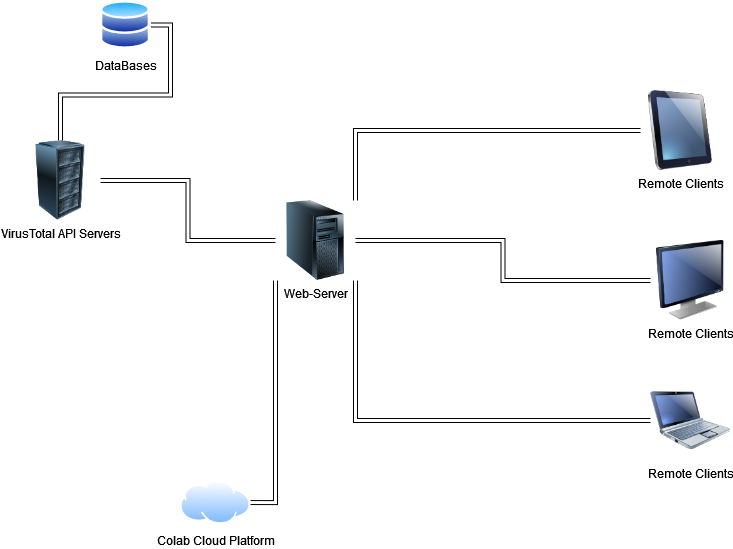
**אתגרים הנדסיים**

**תאימות מערכתית:** הבטחת תאימות למערכות הפעלה שונות יכול להוות אתגר הנדסי, קיימת חשיבות גבוהה לתאימות עם מגוון רחב של מערכות הפעלה על מנת שכלל לקוחות המוצר הפוטנציאלים יוכלו לעשות בו שימוש, וכן שיוכלו לעשות זאת במירב הפלטפורמות בהם הם שתמשים.

**אופטימיזצית מודל האימון:** מציאת סט נתונים אידיאלי, הנדסת מאפיינים, ובחירת אלגורתים למידת מכונה המתאים ביותר וכן כיוונון של ההיפר פרמטרים שלו לקבלת דיוק מירבי דורשת הבנה נרחבת של אלגוריתמי למידת מכונה שונים עבור הערכת חוזקות וחלושות של האלגוריתמים הללו. וכן התבססות על ידע לגבי מודלים שנמצאו כיעילים במחקרים אקדמים שונים.

**זמינות:** יש לידאוג להנדס את הפלטפורמה כך שזמינותה תיהיה גבוה, על כן יש צורך בתכנון ארכיטקטורה שתאפשר זמינות שתעמוד בסטנדרטים האלו, שכן מדובר בתכונה קריטית במיוחד בנוגע לתחום הסייבר.

**חווית משתמש:** יצירת ממשק אינטואיטיבי, ידידותי, ברור, ובטוח למשתמש. חשוב לדאוג על איזון בין העקרונות הללו על מנת לשפר את שביעות רצון המשתמש, וכן להגן על המשתמש מפני הרצה בשוגג של תוכנות זדוניות במהלך השימוש בממשק. רצוי בנוסף שיהיה בהיר למשתמש האם הקובץ הנסרק זוהה כזדוני או נקי.

**ארכיטקטורה:**

**Gantt:**

**תיעוד מרכיבי הפרויקט**

**Utility scripts csv outputs**

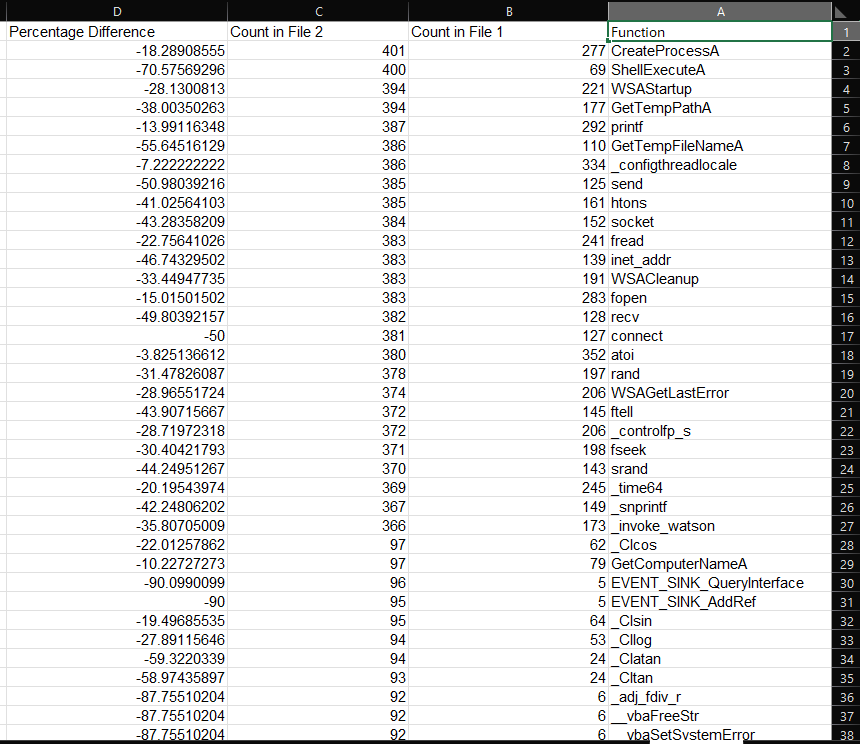
**num of function imports by folder example:**

* A column represents the name of the imported function.
* B column represents the sum of files that imported the respective function.
* this is one of two intermediate csv files that will be an input to the following utility script.

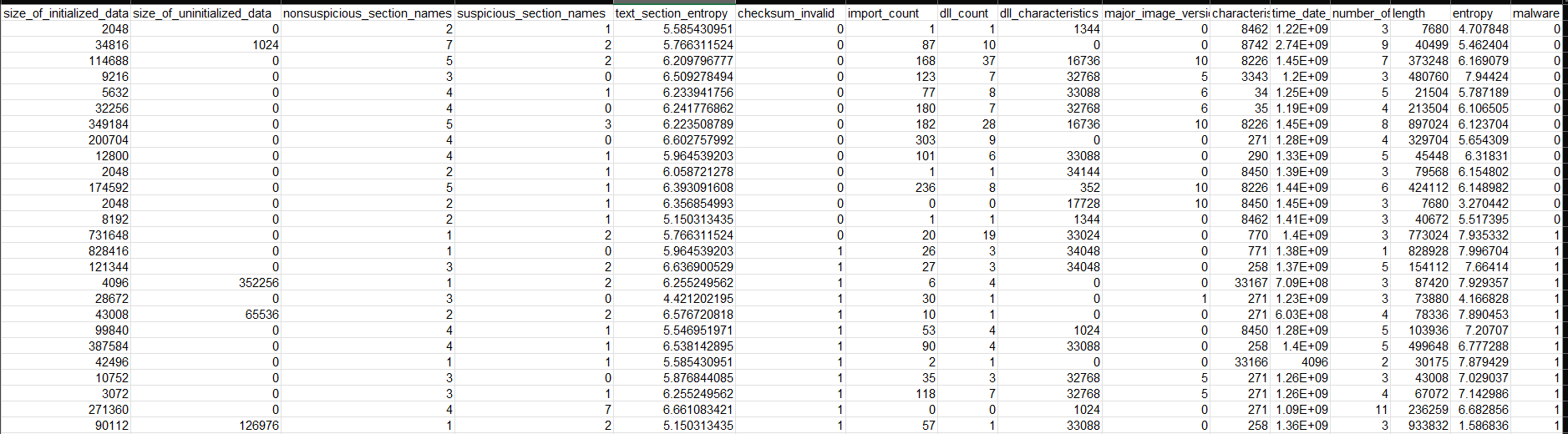
****

**Function import relative difference percentage example:**

* The script receives two ‘num of function imports’ csv files as argument (one that represents the benign files and one that represents the malware files)
* outputs the percentage difference between the two.
* ordered such that the highest sum of import counts from both csv files is at the top.
* can be used to find functions that highly correlate with malwares.

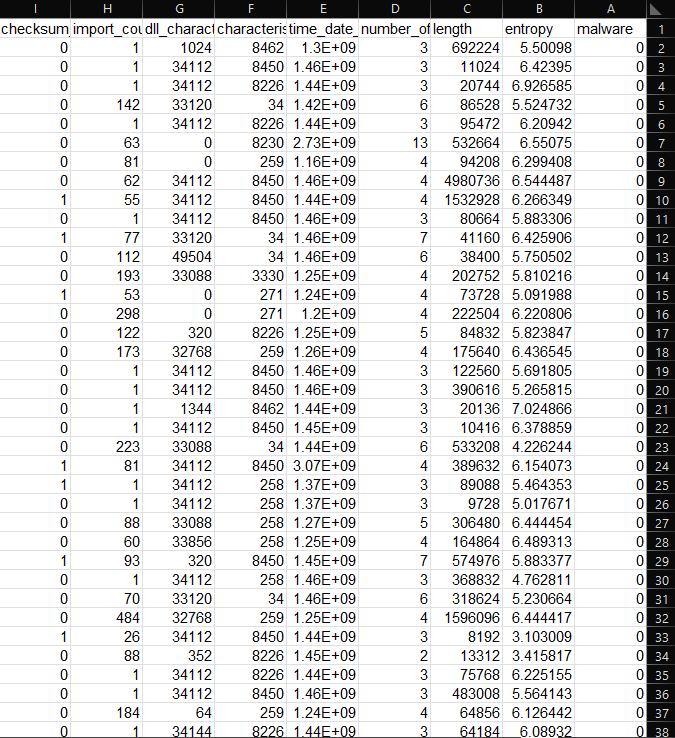
****

**All features extraction dataset generator example:**

* some of the following features have been manually removed as they reduced the performance of the model.

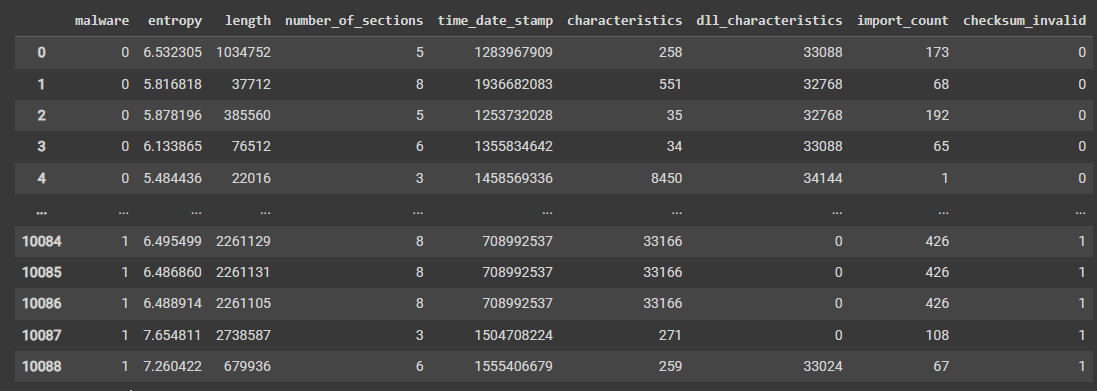
[**https://i.imgur.com/xg9W2TD.png**](https://i.imgur.com/xg9W2TD.png)

**Optimal features extraction dataset generator example**

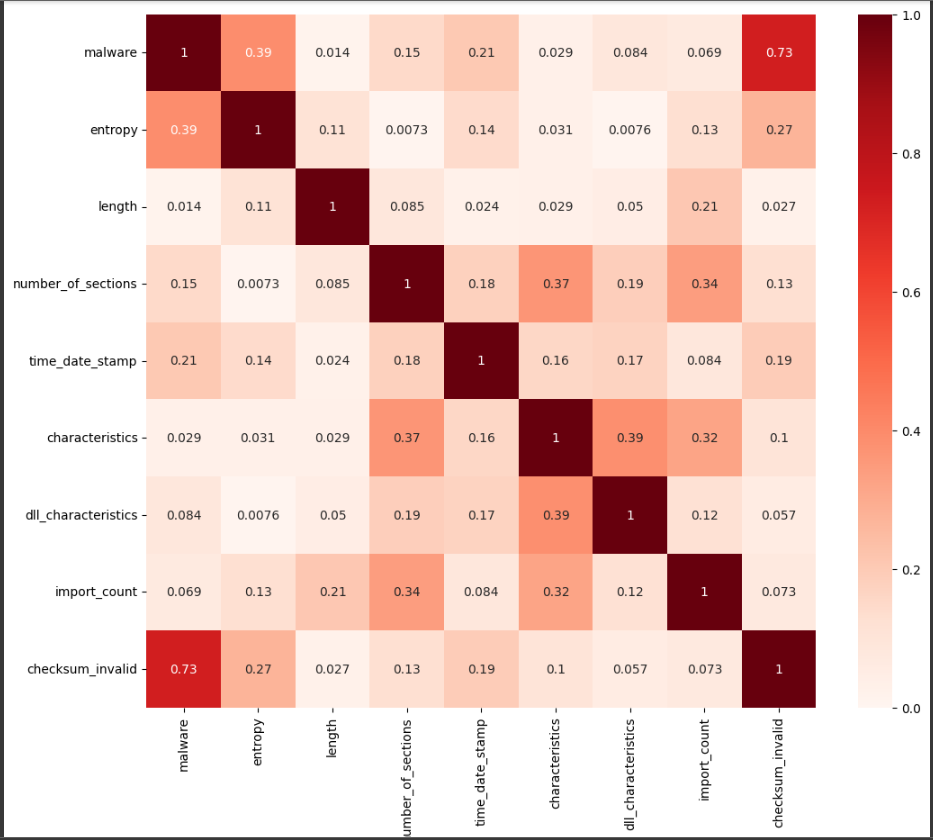
****

**Colab**

**printing dataset dataframe:**

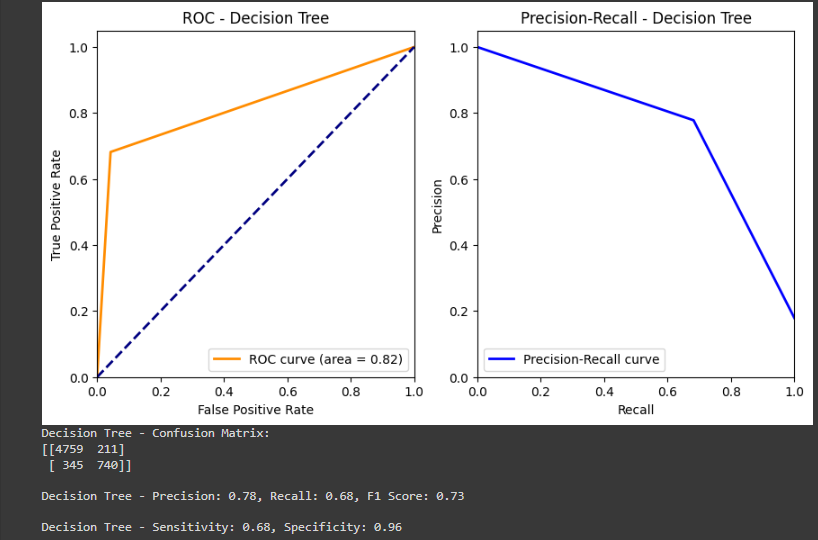
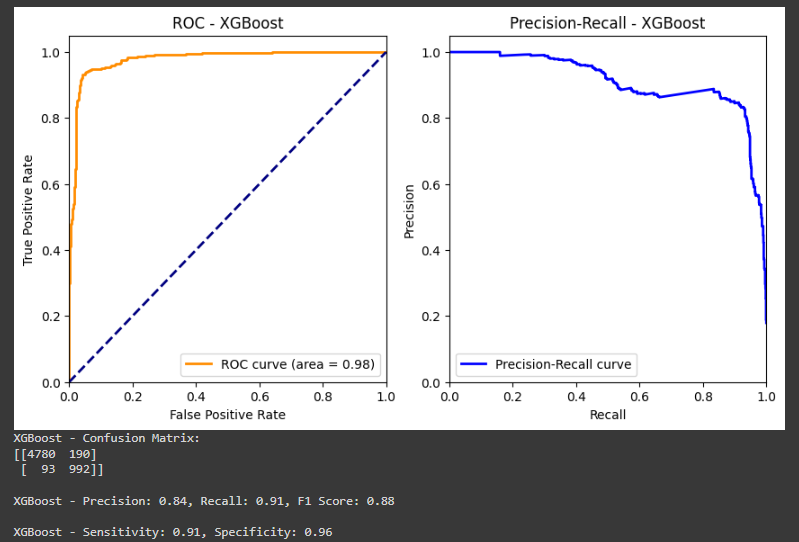
****

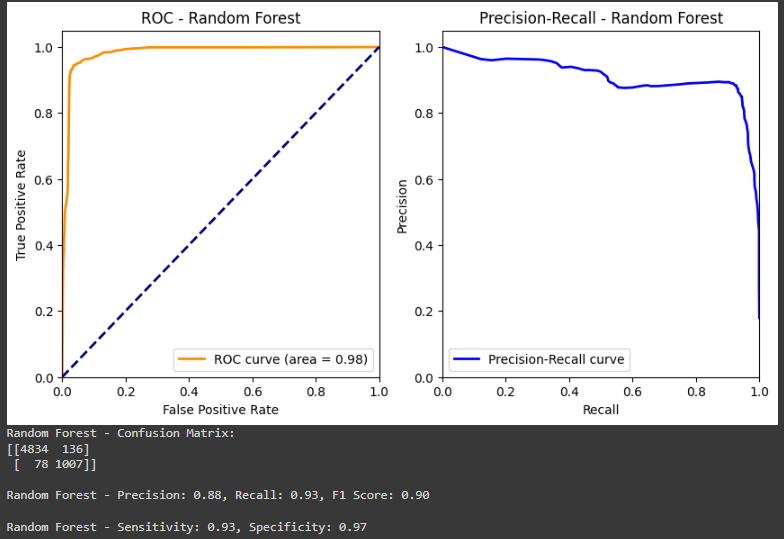
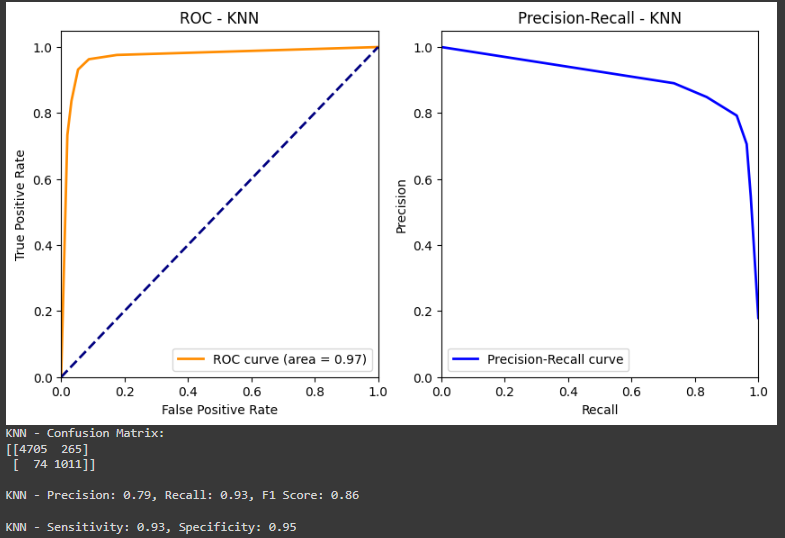
**Correlation matrix visualization of the dataset using heatpmap:**

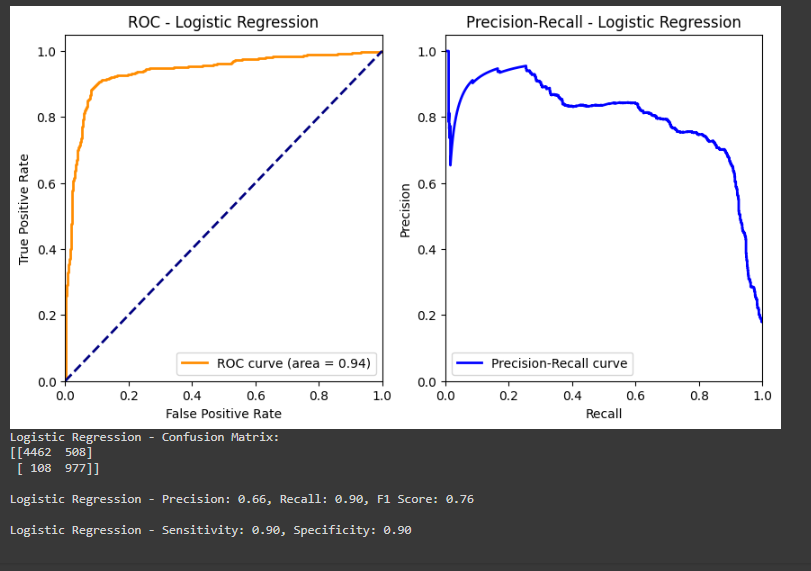
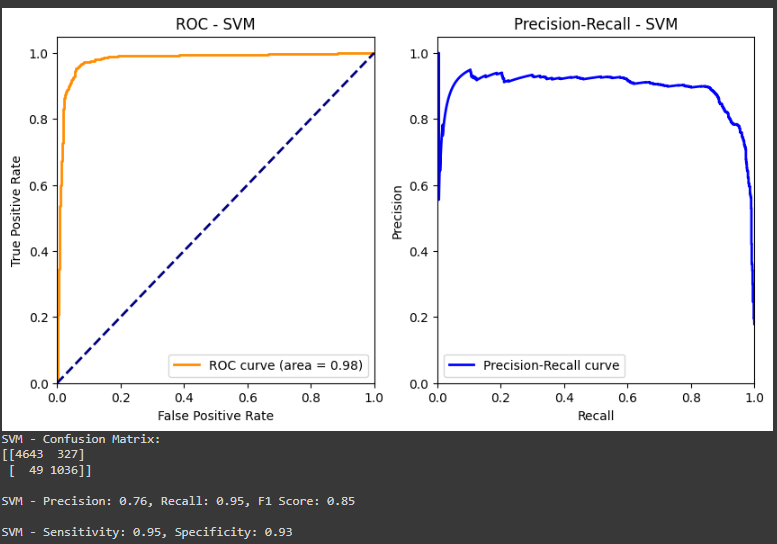
****

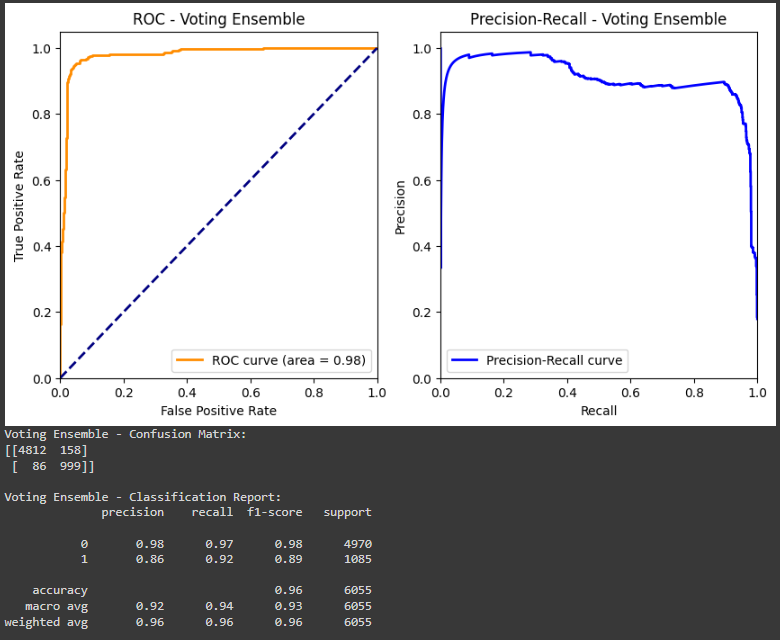
**Multi-Model evaluation post resampling and standard scaling:**

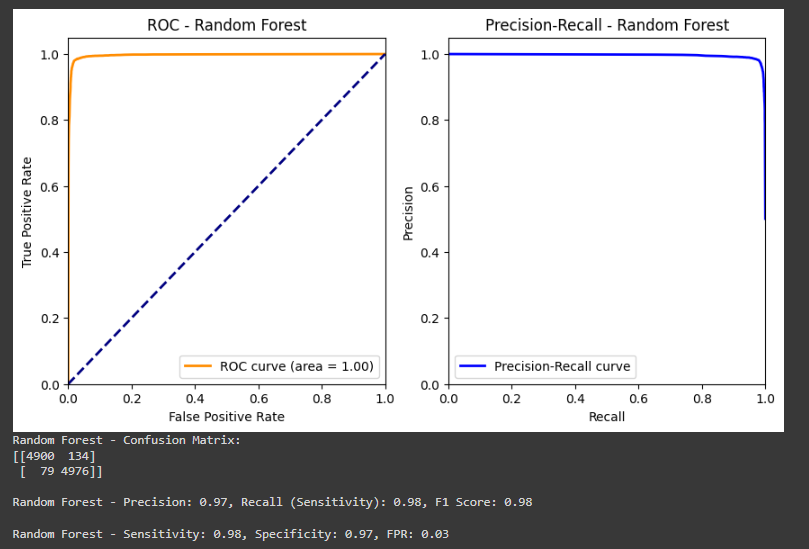
* evaluated on an external test group, not validation splitted from the original dataset.









* cross validation on dataset (without external test)
* Showing only random forest, the best performing model.

ניתן ליראות כי יש עמידה ביעדים על פי המדדים שקבענו, נזכיר כי המדדים הם:

**מדדים:**

* **דיוק זיהוי מעל 70%. (Recall>0.7)**
* פחות מ10% זיהוי שגוי. (FPR<0.1)

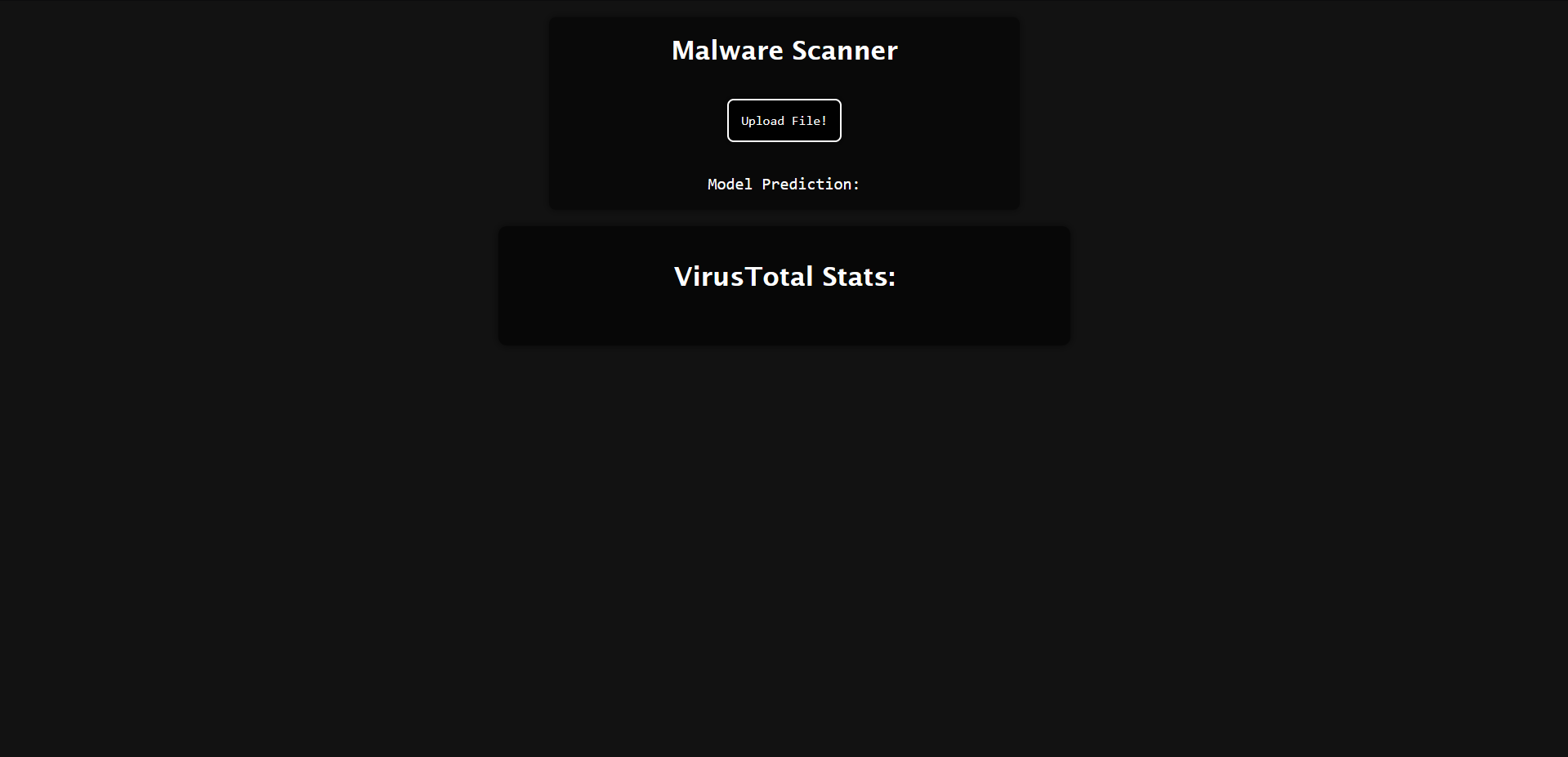
בדיוק הזיהוי (RECALL) קיבלנו 0.98, גבוה בכ40% מהמדד שהצבנו.

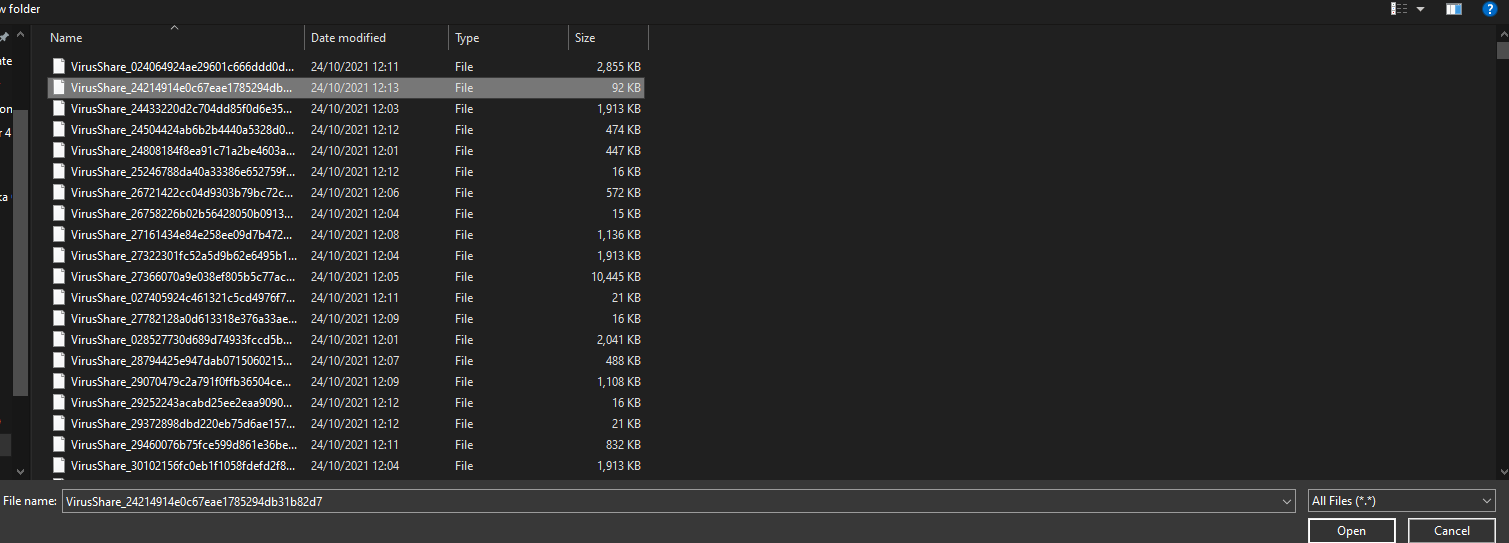
בזיהוי שגוי (FPR) קיבלנו 0.03, נמוך יותר בכ70% מהמדד שהצבנו.

(גם בעבור המבחן החיצוני התקבלו ערכים טובים יותר מהמדדים שהוצבו.)

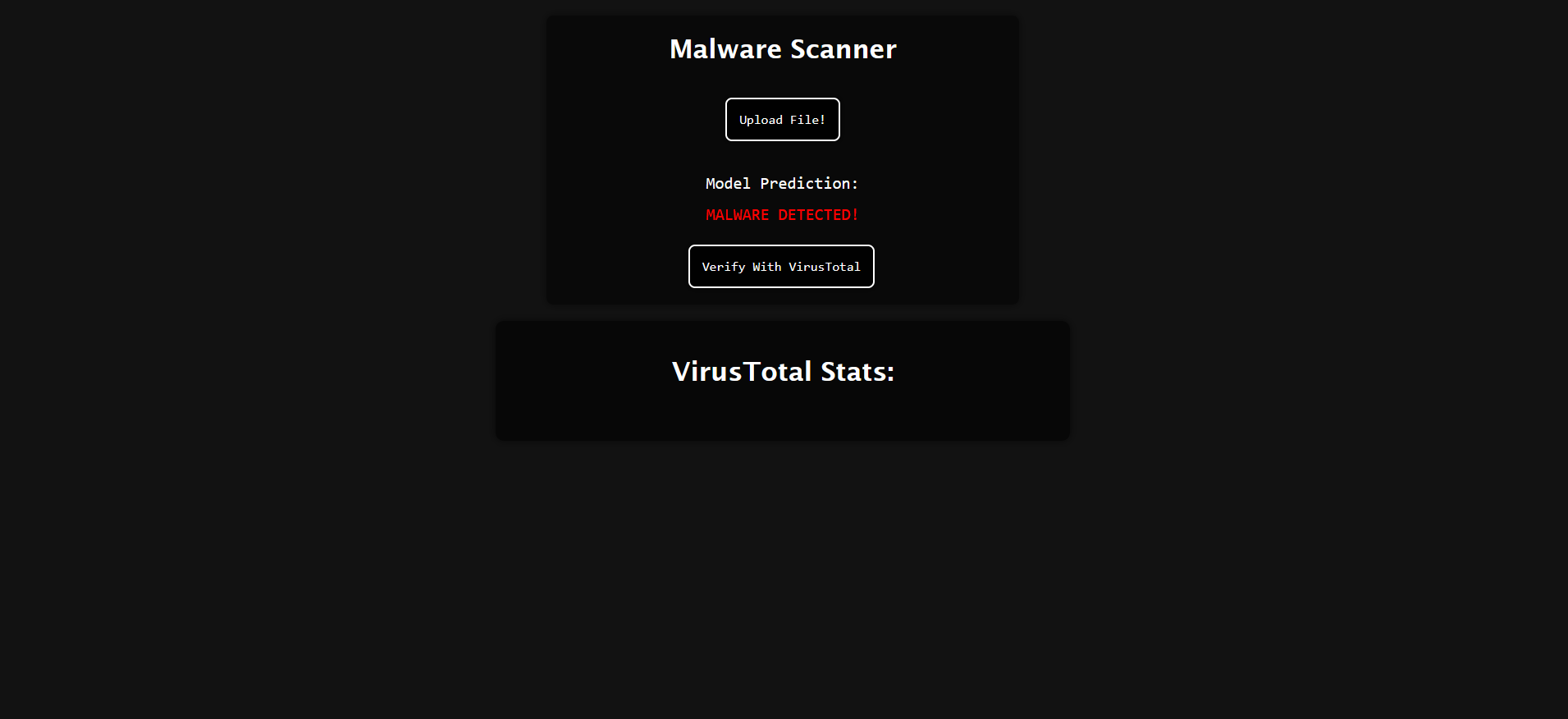
**Website**

**Main page:**

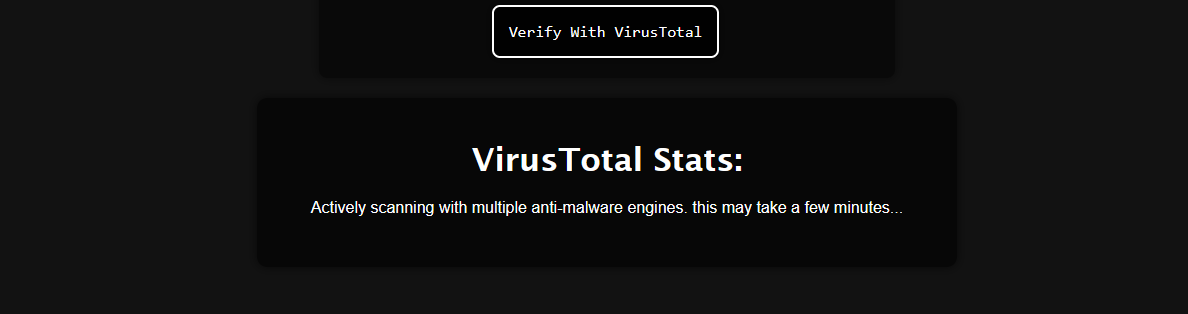
****

**File Selection:**

**Model prediction:**

****

**Awaiting verification by virusTotal:**

****

**VirusTotal stats response:**

****

**דיון**

הפרויקט שלנו נועד לפתח מערכת לזיהוי תוכנות זדוניות תוך שימוש בטכניקות למידת מכונה, בעקבות ולמרות החששות הראשוניים לגבי המורכבות וההיתכנות בהתחשב במשאבי החישוב והידע שלנו, יצרנו קשר עם המרצים הרלוונטים במכללה (ניר אנדלמן, רועי זימון) על מנת לוודא כי נושא הפרויקט אכן רלוונטי עבורנו. ולבסוף, הפרויקט עלה על הציפיות והמדדים, בכך שהפגין דיוק זיהוי גבוה ושיעור חיוביים כוזבים נמוך.

**אתגרים**:

התמודדנו עם מספר אתגרים לאורך הפרויקט, כולל טיפול במערך נתונים לא מאוזן, בו טיפלנו באמצעות SMOTE לצורך איזון על ידי הוספת דגימות מלאכותיות עבור קטגוריית המיעוט. בנוסף, בחירת מאפיינים התבררה כקריטית; בעזרת התבוננות בheatmap ובתוצאות מדדי הערכת החיזוי, התחלנו עם 15 מאפיינים פוטנציאלים וצמצמנו אותם ל-8 ששיפרו משמעותית את ביצועי מודל החיזוי שלנו. מאפיינים אלו כללו אנטרופיה של קבצים, ומספר מאפיינים שנגזרו מכותרת קבצי ההפעלה למשל: מספר מקטעים, חותמת תאריך/זמן והאם הchecksum בתוקף או לא - גורמים שמחקרים הראו כי בעלי קשר חזק לאחת משתי הקטגוריות (זדוני או לא זדוני).

אתגר משמעותי היה מגבלות חישוביות, שדרשו מאיתנו לצמצם את מערך הנתונים הראשוניים שלנו מ-200,000 קבצים לכ-10,000. צמצום זה היה חיוני כדי לנהל את עומס העבודה על משאבי המחשוב האישיים שלנו ולהבטיח ניסויים והכשרת מודלים בזמן המוקצב הנתון לנו.

אתגר משמעומתי נוסף שנתקלנו בו היה ההתייחסות להטיות הטבועות במקור הנתונים. בתחילה, אימון המודל שלנו הופרע על ידי מערך נתונים שנגזר ממקור יחיד, אשר הטתה את ביצועי המודל לעבר תרחישים ספציפיים לאותו מקור. כדי למתן זאת, השקענו מאמצים רבים בגיוון מקורות הנתונים שלנו. זה כלל גישה למאגרי תוכנות זדוניות כמו VirusShare עבורו נידרשנו ליצור קשר עם בעלי האתר לבקשת הרשאות לשימוש עבור הפרויקט, ניצול מערכי נתונים ציבוריים כמו DikeDataset מ-GitHub, ושילוב נתונים מבלוג העוסק בתחום ששמו Practical Security Analytics. על ידי שילוב נתונים ממקורות מגוונים אלה, הצלחנו לבנות סט נתונים מגוון ולכן עוצמתי יותר ששיקף את המציאות, ובכך לשפר את יכולת ההכללה של המודל שלנו על פני סביבות שונות בעולם האמיתי.

יתרה מכך, אתגר נוסף שעמד בפנינו היה הצורך להגביל את הניתוח שלנו לקובצי הפעלה (PE). החלטה זו נבעה בעיקר על ידי אילוצי זמן ותחומי המומחיות הספציפיים שלנו. קבצי PE הם נפוצים ורלוונטיים ביותר בהקשר של תוכנות זדוניות מבוססות ווינדוס.

**תובנות מתודולוגיות**:

המתודולוגיה שלנו כללה לא רק יישום טכניקות למידת מכונה, אלא גם שילוב תובנות ממחקרים קיימים והתאמת מערך המפאיינים שלנו על סמך ניתוח הנתונים מהמחקרים הללו לדוגמה, הכללת תכונת Checksum הושפעה ישירות מהמתאם המוכח שלה עם תוכנות זדוניות בספרות המחקרית.

כמו כן, הדגשנו את החשיבות של מערך נתונים מגוון כדי למנוע הטיות שעלולות להשפיע על יכולת ההכללה של המודל. על ידי מיקור נתונים ממספר מאגרים והבטחת שילוב של סוגי קבצים זדוניים ולא זדוניים שונים, שאפנו ליצור מודל חזק המסוגל לבצע ביצועים טובים על פני תרחישים שונים בעולם האמיתי.

**תהליך אימות:**

כדי להגביר את האמון בתחזיות המודל שלנו, שילבנו אפשרות לאימות תוצאות באמצעות VirusTotal, מה שמאפשר למשתמשים להשוות את הערכות למידת המכונה המהירה שלנו עם ניתוחים יסודיים יותר ממנועים רבים נגד תוכנות זדוניות. שלב זה לא רק הוסיף שכבת אמינות אלא גם סיפק מנגנון המאפשר לבחון קבצים שאינם נתמכים על ידי המודל.

**השקפות על תוצאות:**

התוצאות היו מעודדות ביחס למדדים שהצבנו עבור הפרויקט, המודל היעיל ביותר השיג RECALL של כ 0.98 וFPR 0.03 בבדיקות שלנו. ביצועים אלה מצביעים על כך שאסטרטגיות בחירת המפאיינים ואסטרטגיות האימון שלנו היו מוצלחות לאחר ניסוי וטעייה רבים, במיוחד במקסום היכולת לזהות מקרי תוכנה זדונית אמיתיים תוך מזעור הסיווג השגוי של קבצים שפירים.

**סיכום ומסקנות**

**סקירה כללית של הפרויקט:**

פרויקט זה נועד לפתח מערכת לגילוי תוכנות זדוניות תוך שימוש בלמידת מכונה. לאחר ניסוי וטעייה עם מאפיינים רבים שחילצנו מקבצבים שהבאנו ממקורות שונים בחרנו בכשמונה מאפיינים ובמסווג Random Forest בשל הביצועים האופטימליים שקיבלנו עימם.

**ממצאי מפתח:**

המערכת שלנו השיגה RECALL של 0.98 ו FPR 0.03, תוך עמידה בקריטריוני הביצועים שהצבנו בתחילת הפרויקט. תוצאות אלו מאשרות את האפקטיביות של בחירת המפאיינים שלנו, שבה חידדנו את מספר המאפיינים מחמישה עשר לשמונה הרלוונטים ביותר, כגון אנטרופיה של קבצים ותקפות הCHECKSUM. זה לא רק ייעל את ביצועי המודל אלא גם ייעל את העומס החישובי, דבר הנחשב קריטי בהתחשב במגבלות הזמן והחומרה שלנו.

**משמעות העבודה:**

הפרויקט הוכיח שניתן להשיג דיוק גבוה בזיהוי תוכנות זדוניות על ידי בחירת מאפיינים באופן קפדני ומבוסס, והכשרת מודלים שונים למציאת המודל האופטימלי ביותר, ושניתן לעשות זאת למרות פערי הידע, מגבלות הזמן ומגבלות משאבי החישוב. נוסף לכך, הוכח כי ניתן להשיג באופן חינמי קבצים מקוטלגים ממקורות שונים.

**השקפות אחרונות:**

בהקשפה על הפרויקט, ההצלחה שצפינו מאמתת את נכונות הגישה של שילוב למידת מכונה עם אמצעי אבטחת סייבר מסורתיים. השימוש ב-VirusTotal לאימות מדגיש עוד יותר את מהימנות המערכת שלנו, ומספק מנגנון חיוני לבדיקה כפולה שמשפר את יכולת החיזוי ואת אמון המשתמשים.

**הצעות להמשך עבודה**

**עבודה עתידית:**

במבט קדימה, אנו רואים מספר דרכים לשיפור מערכת זיהוי תוכנות זדוניות שלנו. אלה כוללים שילוב מאפיינים סטטים נוספים, מאפיינים שיחולצו מניתוח דינמי, וניסיון שימוש באלגוריתמים אחרים של למידת מכונה שעשויים להציע דיוק או יעילות משופרים. יתר על כן, עדכון מתמשך של המודל עם דוגמאות תוכנות זדוניות חדשות וקבצים שפירים יהיה חיוני כדי לשמור על הרלוונטיות והיעילות שלו נגד איומי סייבר מתפתחים.

**מקורות**

[[1]](https://www.researchgate.net/publication/229008321_A_survey_of_malware_detection_techniques) Nwokedi Idika and Aditya P. Mathur ,“A Survey of Malware Detection Techniques”

[[2]](https://abuisa.github.io/reff/Classification-of-malware-families-based-on-N-grams-sequential-pattern-features.pdf) Chatchai Liangboonprakong and Ohm Sornil, “**Classification of Malware Families Based on N-grams Sequential Pattern Features”**

[[3]](https://papers.put.as/papers/ml/2014/markel2014.pdf) Zane Markel and Michael Bilzor, **“Building a Machine Learning Classifier for Malware Detection”**

[[4]](https://www.researchgate.net/publication/338486766_Feature_Selection_Using_a_Machine_Learning_to_Classify_a_Malware) Mouhammd Al-Kasassbeh, Safaa Mohammed, Mohammad Alauthman, and Ammar Almomani**,** ” **Feature Selection Using a Machine Learning to Classify a Malware”**

[[5]](https://en.wikipedia.org/wiki/Decision_tree_learning) Wikipedia, **“Decision tree learning”**

[[6]](https://cobweb.cs.uga.edu/~liao/PE_Final_Report.pdf) Yibin Liao, **“PE-Header-Based Malware Study and Detection”**

[[7]](https://scholarworks.sjsu.edu/cgi/viewcontent.cgi?article=1624&context=etd_projects) Samuel Kim, **“PE Header Analysis for Malware Detection”**

[[8]](https://www.sid.ir/FileServer/SE/648E20200653.pdf) Tina Rezaei, Ali K. Hamze **“An Efficient Approach For Malware Detection Using PE Header Specifications”**