**סקירת ספרות – פרוייקט פיתוח  
Auto-Defense ML**

תוכן עניינים

[**תקציר** 2](#_Toc158641720)

[**חקר מצב קיים** 3](#_Toc158641721)

[**תיאור הלקוח** 3](#_Toc158641722)

[**מערכות קיימות** 4](#_Toc158641723)

[**ספריות דומות ל ART** 4](#_Toc158641724)

[**מערכות דומות לפלטפורמת Auto-Defense ML** 4](#_Toc158641725)

[**סקירת טכנולוגיות רלוונטיות** 6](#_Toc158641726)

[**ספריות** 6](#_Toc158641727)

[**אלגוריתמיקה** 6](#_Toc158641728)

[**Multi-Criteria Decision Making(MCDM)** 6](#_Toc158641729)

[**Technique for Order Preference by Similarity to Ideal Solution(TOPSIS)** 7](#_Toc158641730)

[**חקר דרישות ואפיון ראשוני** 8](#_Toc158641731)

[**מטרת המערכת** 8](#_Toc158641732)

[**סיכום תהליך חקר הדרישות** 8](#_Toc158641733)

[**תהליכים עסקיים** 9](#_Toc158641734)

[**דרישות פונקציונאליות** 14](#_Toc158641735)

[**דרישות לא פונקציונאליות** 14](#_Toc158641736)

[**תכנון המנשק** 15](#_Toc158641737)

[**שימוש בנתונים** 15](#_Toc158641738)

[**תוצרים מתוכננים** 15](#_Toc158641739)

[**אתגרים וסיכונים** 15](#_Toc158641740)

[**כלים ושיטות** 15](#_Toc158641741)

[**לו''ז מעודכן לפרוייקט** 15](#_Toc158641742)

[**חלוקת אחראיות** 15](#_Toc158641743)

[**ביבליוגרפיה** 17](#_Toc158641744)

# **תקציר**

מערך הסייבר הלאומי הינו גוף ממשלתי האחראי על אבטחת מידע והגנת סייבר ברמה הלאומית של מדינת ישראל.  
תפקידיה כוללים הובלת מאמצי הגנה על מרחב הסייבר הלאומי, חוסן המשק, מובילות וחדשנות וקידום אסטרטגיות ומדיניות בתחום הסייבר, ולהוות מוקד ידע וסמכות מקצועית וגוף מייעץ לממשלה.  
  
מעבדת Offensive AI באונ' בן גוריון חוקרת כיצד ישויות זדוניות משתמשים בבינה מלאכותית כדי לשפר את ההתקפות שלהם. מטרת המעבדה היא להגן על החברה על ידי זיהוי והפחתה של איומי AI לפני שהם הופכים למיינסטרים.  
  
במסגרת מעבדת Offensive AI באוניברסיטת בן-גוריון מפותחת פלטפורמת Auto-Defense ML  
עבור מערך הסייבר הלאומי, בשיתוף עם גוגל.  
מטרת הפלטפורמה הינה הערכת ההגנה של מודלי למידת מכונה נגד התקפות שונות, באופן אוטומטי ונוח למשתמש.  
המהנדס הממוצע אינו מקצוען, הוא יכול בקלות לבנות מודלי למידת מכונת, אך אינו בהכרח מתמקצע במאחורי הקלעים, או מודע לסכנות הטמונות של המודלים שהינו מפתח.  
בכך, הפלטפורמה מאפשרת באמצעות point and click קל ונוח לבצע הערכה, ולהציע למשתמש הגנות מתאימות.  
  
הייחודיות של הפלטפורמה אינה רק העובדה שמדובר בשימוש קל, אלא גם כן בהוצאת דו''ח מפורט הכולל מידע על כלל ההתקפות וההגנות שבוצעו, המלצות להגנה מתאימה ו pipeline שמבצע אוטומציה ברמה שלא קיימת במערכות אחרות. בעוד שבמערכות אחרות מגדירים ידנית התקפה או הגנה, מגדירים מטריקות, וכל דבר אחר שדרוש ממקצוען לעשות, המערכת שלנו באופן אוטומטי תוכל כבר לענות על השאלות הבאות:

1. איך המודל מתפקד ללא הגנות וללא התקפות?
2. איך המודל מתפקד בעת ביצוע התקפה ללא הגנה?
3. איך המודל מתפקד בעת מימוש הגנה ללא התקפה?
4. איך המודל מתפקד בעת מימוש הגנה וביצוע התקפה?

בנוסף לכך, כלל הפריימוורקס אשר מממשים התקפות והגנות שונות, אינם תומכים במימושים ספציפיים עבור מודלים של עצי החלטה וensemble, על כך במסגרת הפרוייקט אנחנו נממש התקפות והגנות אלו, כך שהפלטפורמה תתמוך ביותר מודלים ותבצע הערכה יותר טובה.

# **חקר מצב קיים**

## **תיאור הלקוח**

בעוד שבמסגרת הפרוייקט משתתפים ארגונים שונים באופן ישיר ועקיף בחלקים השונים שלו, הלקוח העיקרי שלנו הינו מערך הסייבר הלאומי.  
  
במסגרת מעבדת Offensive AI באונ' בן-גוריון, מפותחת פלטפורמה בשם Auto-Defense ML עבור מערך הסייבר הלאומי. כמו כן חברה שם Kaleido עוזרת בפיתוח כחברת צד-שלישי.  
מטרת הפלטפורמה הינה אוטומציית טסטים לצורך זיהוי בעיות וקבלת הצעות לחיזוק מודלי למידת מכונה.  
הפלטפורמה משתמשת בספריית ART לביצוע ההתקפות וההגנות בפועל.  
ספריית ART(Adversarial-Robustness-Toolbox) הינה ספריית קוד-פתוח אשר מנוהלת על ידי IBM וממומנת על ידי ארגון DARPA. הספרייה תומכת במגוון מאוד רחב של ML Frameworks.  
  
בעוד שהפלטפורמה מבצעת את האוטומצייה, יש בעיות קיימות שהמערכת צריכה לפתור, וכאן נכנס לתמונה הפרוייקט שלנו, שבעצם מבצע הרחבה לפלטפורמה זו.  
בפלטפורמה קיימים מס' בעיות עיקריות שהמערכת שלנו תביא לפתרון:

1. הפלטפורמה אינה יוצרת דו''ח מפורט, בעל וויזואליזציה, והמלצות עבור המשתמש.  
   אנו נרצה לבנות מודול האחראי ליצירת הדו''ח, אשר יהיה קל לעשות לו אינטגרציה ביחד עם הפלטפורמה של המעבדה.
2. הפלטפורמה אינה תומכת ב GUI.  
   כדי לבצע התקפות והגנות יש צורך לתכנת ו/או השימוש הוא ב CLI.  
   מטרתינו יהיה לספק GUI כך שהפעולות יהיו קלות ונוחות עבור המשתמש.
3. בספריית ART לא קיים מימוש ספציפי של התקפות והגנות עבור עצי החלטה(XGBoost)  
   נממש הגנות והתקפות אלו אשר הינם ספציפיים עבור מודלי עצי החלטה ,המימושים יבוצעו במסגרת ART, כלומר כתרומת קוד-פתוח לספרייה, ותבוצע אינטגרציה ל pipe-line של הפלטפורמה.

## **מערכות קיימות**

קיימים מס' מערכות/פלטפורמות/פריימוורקס דומות למערכת או ל ART עצמו, לכן נדגיש את הפופולריות ביותר. כמו כן נחלק סעיף זה לשניים, נשווה ספריות ל ART ולפלטפורמה.

### **ספריות דומות ל ART**

1. **Torch attacks**  
   *https://github.com/Harry24k/adversarial-attacks-pytorch*הספרייה מספקת מימוש להתקפות ב interface דומה ל PyTorch, ולכן מכוונת בעיקר אליה, בנוסף מס' ההתקפות מאוד נמוך ואינו כולל התקפות ספציפיות עבור עצי החלטה.  
   כמו כן קיימת ספרייה torch defenses אך היא עדיין בשלבי בנייה.
2. **Foolbox***https://github.com/bethgelab/foolbox*ספריית Foolbox תומכת רק בשלושה פריימוורקים של למידות מכונה PyTorch,TensorFlow,JAXקיים לה מגוון יותר רחב של התקפות מאשר torch attacks, אך לא קיים מימוש להגנות, הספרייה מתמקדת בהתקפות בלבד, ואינה כוללת התקפות ספציפיות עבור עצי החלטה.

### **מערכות דומות לפלטפורמת Auto-Defense ML**

1. **Microsoft counterfit**  
   https://github.com/Azure/counterfit  
   מערכת קוד-פתוח ע''י מיקרוסופט, משתמשת בהתקפות מ ART .  
   בנוסף המערכת תומכת ב TextAttack אשר הינה ספריית התקפות עבור מודלי NLP.  
   המערכת מבצעת התקפות באופן פשטני דרך ה CLI אך אינה מבצעת הגנות.  
   כלומר ה pipeline הינו מאוד פשטני: בחירת התקפה, בחירת מודל וביצוע התקיפה.  
   בניגוד למערכת שלנו, אין GUI , ה pipeline מאוד פשוט ולא כולל הגנות, לא קיים ייצוא דו''ח מפורט עם המלצות, אך היתרון הינה כי תומכת בנוסף ב TextAttack.
2. **Armory**  
   https://github.com/twosixlabs/armory-library  
   ארמורי הינה ספריית קוד-פתוח test bed אשר מבצעת הערכה למתקפות.  
   הספרייה משתמשת בהתקפות מ ART.  
   ניתן להגדיר מטריקות וליצור קונפיגורציות ובכך להריץ הערכה של התקפה על מודל.  
   השימוש בה הינה תכנותית בלבד.  
   החסרונות דומים ל countefit בכך שאין GUI, לא קיים ייצוא דו''ח עם המלצות, וה Pipeline לא כולל הגנות אך היתרון הינם שימוש בקונפיגורציות.
3. **Cleverhans**  
   https://github.com/cleverhans-lab/cleverhans  
   ספריית פייתון קוד-פתוח להערכת ההגנה של מודלי למידת מכונה.  
   תומכת בשלושה פריימוורס בלבד : TF2,PYTORCH,JAX  
   מימוש ההתקפות וההגנות נמצא בספרייה עצמה ולא דרך ספרייה צד-שלישי.  
   השימוש בה הינה תכנותי, ובניגוד ל armory ששם ניתן בקלות להגדיר קובץ קונפיגורציה, בספרייה זו יש לעשות הכל ידנית, ואין אוטומציה משום צורה כלל.  
   החסרונות הם כנ''ל כמו למערכות האחרות.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DT Support | # of natively supported ML frameworks | # Defenses | # Attacks |  |
| Only 1 attack for scikit models. | **All popular frameworks(8+)** | **39** | **83** | **ART** |
| None | **1** | **None** | **37** | **Torch attacks** |
| None | **3** | **None** | **48** | **Foolbox** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ML Framework support | Output | Workflow | Usage of attacks&defenses | User Interaction |  |
| All | Detailed report of all possible metrics with visualizations. | Automatic evaluation of overall security with and without defenses. | ART | GUI | **AutoDef-ML** |
| All | Selected specific metrics | Only selected attacks | ART,TEXTATTACK | CLI | **Counterfit** |
| All | Selected specific metrics | Only selected attacks | ART | Programming | **Armory** |
| TF2,PYTORCH,JAX | Selected specific metrics | Only selected attacks and defenses | Self-implemented | Programming | **Cleverhans** |

# **סקירת טכנולוגיות רלוונטיות**

כלל המערכת תפותח בשפת פייתון, כאשר אנחנו נותנים דגש על ספריות שהינם חינמיות וקוד-פתוח.

## **ספריות**

* **Adversarial-robustness-toolbox** בבקאנד לביצוע ומימוש התקפות והגנות.  
  השימוש בספרייה זו היא דרישה עיקרית ואינה בחירה, שכן הפלטפורמה שאנחנו מרחיבים משתמשת בה, ונרצה לשמור על קונסיסטנטיות, ובנוסף לתרום מימוש משלנו עבור התקפות והגנות לעצי החלטה. כמו כן כפי שפורט בחלק הקודם, ספרייה זו היא המתקדמת ביותר בתחום נכון לעכשיו ותומכת במספר הרב ביותר של ML Frameworks ומאוד נוחה לשימוש.
* **Dearpygui** למימוש ה GUI.  
  ספריות פופולריות ליצירת GUI כוללות את tkinter,pyqt,kivy אך dearpygui  
  הינה הבולטת שכן היא מציעה נוחות שימוש עם דגש על פשטות וביצועים.  
  בנוסף הספרייה תומכת ב cross-platform.
* **ReportLab & matplotlib**ספריית ReportLab מסוגלת לייצר דו''חות PDF מורכבים בעלי טקסט ו matplotlib מסוגל ליצור גרפים לצרכי ויזואליזציה ולהכניס אותם ישירות ל pdf.  
  בנוסף לכך, התוצר בפועל של המערכת שלנו הינו קובץ json אשר יכלול את כלל הנתונים והתוצאות, ובספריית reportlab קיים מודול json2pdf אשר יעזור לנו במטלה זאת.  
  ספריות אחרות כגון pyfpdf הינם מאוד מופשטים ואינם מתאימים למטרה שלנו.
* **PyInstaller**לביצוע packging ויצירת installer נשתמש בספריית PyInstaller.  
  קיימים הרבה ספריות אחרות, כאשר ברובם כל ספרייה מתמקדת בפלטפורמה מסויימת.  
  PyInstaller יאפשר לנו ליצור standalone executable עבור כל פלטפורמה בצורה נוחה.
* **SHAP** לצרכי XAIבמקרים מסויימים תהיה לנו האפשרותלהשתמש ב SHAP על מנת להגביר את ה interpretability של התוצאות. נציין כי בשלב זה מדובר עדיין רק ברעיון שאנחנו חוקרים.

## **אלגוריתמיקה**

בעוד שה Pipeline הראשי שלנו ישתמש באלגוריתם שאנחנו נממש בעצמינו, מהסיבה שלא מדובר במשהו מורכב אלה בסדרת פעולות. מערכת ההמלצה שלנו תהיה תלויה בתוצאות מטריקות הרבות שנקבל כגון accuracy,precision,recall וכו'. אזי, מתבקשת השאלה, איך נדרג את ההערכה הטובה ביותר ובכך נמליץ למשתמש הגנה טובה ביותר, אם בכלל, שכן יישום הגנה בהכרח פוגעת במודל עצמו ?  
כלומר, אנחנו נכנסים לתחום שנקרא Multi-Criteria Decision Making(MCDM).

### **Multi-Criteria Decision Making(MCDM)**

MCDM הוא תחום קריטי במדעי ההחלטות, רלוונטי במיוחד כאשר החלטות חייבות להיעשות בנוכחות קריטריונים מרובים, לעתים סותרים. תחום זה רלוונטי במיוחד בהערכת מודלים של למידת מכונה שבהם יש לשקול מדדי ביצועים שונים (כמו דיוק או ציון F1) בו זמנית כדי לקבל החלטות מושכלות לגבי בחירת מודל, דירוג או זיהוי אמצעי ההגנה הטובים ביותר.

MCDM מספקת גישה מובנית לקבלת החלטות בכך שהיא מאפשרת הערכה והשוואה של חלופות שונות המבוססות על מספר קריטריונים. זה כרוך בהקצאת משקלים לקריטריונים שונים על סמך חשיבותם, ולאחר מכן שימוש במשקלים אלה לחישוב ציון מורכב עבור כל חלופה. לאחר מכן, החלופה עם הניקוד הטוב ביותר נבחרה כבחירה האופטימלית.  
  
קיימים הרבה אלגוריתמים בתחום זה, אנו בחרנו באלגוריתם בשם TOPSIS משום שאלגוריתם זה הינו יחסית פשוט, קל להבנה ומימוש ויעיל – הרצת התקפות והגנות הינו תהליך שלוקח זמן כבר ככה.  
סיבה עיקרית לבחירת אלגוריתם זה הינו העובדה שיהיה קל להכניס קריטריון robustness ולהביא לו משקל גבוהה ביחס לקריטריונים אחרים, שכן אנחנו יודעים כי הערכה על מודל נקי כנראה תביא את התוצאה הגבוהה ביותר, ומודל בעל הגנה בהכרח יוריד מביצועי המודל, וTOPSIS בעל משקול גבוה ל robustness יכול לעזור לנו להתגבר על בעיה זו.

### **Technique for Order Preference by Similarity to Ideal Solution(TOPSIS)**

TOPSIS היא שיטת קבלת החלטות מרובת קריטריונים (MCDM) המדרגת חלופות על סמך המרחקים שלהן מהפתרון האידיאלי ומהפתרון השלילי-אידיאלי. הפתרון האידיאלי הוא החלופה ההיפותטית בעלת הערך הטוב ביותר עבור כל קריטריון שנחשב, בעוד שלפתרון השלילי-אידיאלי יש את הגרוע ביותר. TOPSIS מחשב את המרחק האוקלידי של כל חלופה לשני פתרונות אלה וקובע מקדם קרבה, המייצג עד כמה כל חלופה קרובה לפתרון האידיאלי ביחס לפתרון השלילי-אידיאלי. החלופה עם מקדם הקרבה הגבוה ביותר נחשבת לבחירה הטובה ביותר. TOPSIS נמצא בשימוש נרחב בשל הפשטות, המובנות והיכולת שלו להתמודד עם תרחישי קבלת החלטות מורכבים הכוללים מספר קריטריונים.

המאמר הבסיסי ל-TOPSIS נכתב על ידי צ'ינג-לאי הוואנג וקוואנגסון יון ושמו "Multiple Attribute Decision Making: Methods and Applications". הוא פורסם בסדרת Springer in Operations Research בשנת 1981. עבודה זו הניחה את הבסיס לשיטת TOPSIS ויישומה בהקשרים שונים של קבלת החלטות.  
  
**יתרונות**:

* **אינטואיטיבי**: קל להבין ולהסביר את הרעיון של בחירת חלופות הקרובות ביותר לאידאל והרחוקות ביותר מהאידיאל השלילי.
* **צדדיות**: ניתן ליישם את TOPSIS על פני תחומים שונים ועם סוגים שונים של נתונים.
* **יעילות**: השיטה פשוטה מבחינה חישובית, ומאפשרת דירוג מהיר של חלופות.

**חסרונות:**

* **משקלים יחסיים:** התוצאות תלויות במידה רבה במשקלים המוקצים לקריטריונים, שיכולים להיות סובייקטיביים ועשויים שלא תמיד לתפוס את החשיבות האמיתית של כל קריטריון.
* **הנחה של עצמאות קריטריונים**: TOPSIS מניח שהקריטריונים הם בלתי תלויים, מה שעשוי לא להתקיים בכל ההקשרים של קבלת החלטות.
* **השפעת חריגים**: השיטה יכולה להיות רגישה לחריגים, שכן ערכים קיצוניים יכולים להשפיע באופן משמעותי על הפתרונות האידיאליים והשליליים-אידיאליים.

המסקנות העיקריות מהעבודה של הוואנג ויון על TOPSIS מדגישות את היעילות והמעשיות שלה בפתרון בעיות MCDM. הם מדגימים ש-TOPSIS מספקת מסגרת ברורה והגיונית לדירוג חלופות המתאימות לתהליכי קבלת החלטות אינטואיטיביים. עם זאת, הם גם מכירים בחשיבות של בחירת משקלי קריטריונים בקפידה ובצורך בהבנה מקיפה של הקשר ההחלטה כדי להבטיח דירוג מדויק ומשמעותי.

# **חקר דרישות ואפיון ראשוני**

## **מטרת המערכת**

מטרת המערכת היא לפתח פלטפורמה אוטומטית שמעריכה מודלים של למידת מכונה, במיקוד על עצי החלטה, בתנאים שונים, כולל פעולה רגילה, תחת התקפה ללא הגנות ותחת התקפה עם מנגנוני הגנה שונים מיושמים.

* **הערכת ביצועי מודל**: הערכת ביצועים בהיעדר התקפות באמצעות מדדים סטנדרטיים כגון דיוק ו precision המספקים הבנה בסיסית של היעילות של כל מודל.
* **ניתוח פגיעות להתקפות**: כיצד סוגים שונים של התקפות משפיעות על ביצועי המודל, זיהוי נקודות תורפה וחולשות פוטנציאליות במודלים ללא מנגנוני הגנה כלשהם.
* **הערכת מנגנוני הגנה**: היעילות של אסטרטגיות הגנה שונות על ידי הערכת ביצועי המודל בתנאי התקפה כאשר הגנות אלו מיושמות, עוזר להבין עד כמה כל הגנה מגנה על המודל.
* **איזון ביצועים וחוסנות**: trade-off בין שמירה על ביצועים גבוהים על נתונים נקיים (ללא תרחישי תקיפה) לבין השגת חוסן מפני התקפות, תוך הבטחה שהדגם או מנגנון ההגנה הנבחרים מציעים איזון אופטימלי.
* **אוטומציה של קבלת החלטות**: שימוש באלגוריתמים כמו TOPSIS כדי לצבור תוצאות מההערכות ולדרג באופן שיטתי ולהציע את מנגנוני ההגנה או תצורות המודל הטובות ביותר. זה כרוך בבחינת קריטריונים מרובים (כולל מדדי ביצועים וציוני חוסן) כדי לקבל החלטות מושכלות.
* **אמון ושקיפות:** סיפוק תובנות ברורות וניתנות לפירוש מדוע מומלצות הגנות מסוימות, תוך שיפור האמינות והקבילות של המלצות המערכת.

## **סיכום תהליך חקר הדרישות**

הפרויקט שלנו התחיל בחקירה מקיפה של Advarsarial Machine Learning, תחום קריטי שבו מוערכים מודלים של למידת מכונה מול איומי אבטחה פוטנציאליים. שלב זה היה חיוני לביסוס הבנה בסיסית של טרמינולוגיות, מושגים וטקטיקות התקפה והגנה.

חלק נכבד מהמחקר הראשוני שלנו כלל התעמקות בתיעוד הנרחב שסופק על ידי ART.  
זה עזר לנו לתפוס את היכולות הפונקציונליות של ART, כולל סימולציות תקיפה ומנגנוני הגנה. התיעוד שימש כמקור הידע הטכני העיקרי שלנו, והנחה את ההבנה שלנו כיצד ניתן למנף את ART כדי לדמות תנאים בעולם האמיתי ולהעריך את החוסן של מודלים של למידת מכונה.  
הצוות שלנו עסק בניסויים מעשיים עם ART כדי לממש את ההבנה התיאורטית שלנו. המעורבות המעשית הזו הייתה מרכזית בפענוח המורכבות של הארכיטקטורה של ART, לרבות ממשקי ה-API שלה, והשילוב של מודולי התקפה והגנה שונים.

**מסקנות עיקריות:**

* המחקר שלנו האיר את המערך העצום והמורכב של התקפות שניתן לפרוס נגד מודלים של למידת מכונה, והדגיש את הצורך הקריטי באסטרטגיות הגנה חזקות.
* ART מתגלה כמשאב רב ערך להדמיית תרחישים זדוניים, ומציע מגוון רחב של התקפות והגנות מובנות מראש המאפשרות הערכות אבטחה מקיפות.
* ידע תיאורטי של למידת מכונה אדוורסרית, אמנם יסודי, אך יש להשלים עם ניסויים מעשיים כדי להבין באופן מלא את הדינמיקה של התקפות והגנות.
* תחום למידת המכונה האדוורסרית מתפתח במהירות, ומצריך מחקר מתמשך והתאמה לאיומים ואמצעי נגד חדשים.

## **תהליכים עסקיים**

התהליכים העסקיים שלנו כוללים ארבעה תהליכים עיקריים, שאותם נתאר בעזרת פסאודו-קוד.

1. איסוף המידע שהמשתמש נתן מה GUI , כגון קוניפוגרציה, מודל, ונתונים נוספים.
2. ביצוע התקפות והגנות והערכה- ה Pipeline הראשי שלנו.
3. דירוג והמלצה.
4. יצירת הדו''ח.





**UC1: Deploy a Model**

**Preconditions:**

The user has a model file containing all the weights, parameters, and architecture, in a compatible format (e.g., .h5 for Keras models, .pt for PyTorch).

**Postconditions:**

The model file is successfully saved within the system.

**Main Success Scenario:**

1. The user logs into the system.
2. The user navigates to the 'Model Management' section.
3. The user clicks the 'Upload Model' button.
4. The system prompts the user to select a model file from their device.
5. The user selects the correct model file and clicks 'Open'.
6. The system validates the file format and compatibility.
7. The file is successfully uploaded and saved within the system.

**Extensions:**

3a. The user decides to view the existing models before uploading a new one.

3a1. The system displays the list of previously uploaded models.

**Alternative Flows:**

6a. The selected file does not meet the required format or specifications.

6a1. The system displays an error message: "Unsupported file format. Please upload a compatible model file."

6a2. The user is prompted to select another file.

**UC2: Choose an Attack/Defense Type**

**Preconditions:**

The user has successfully deployed a model within the system.

The user is in the 'Attack/Defense Configuration' section.

**Postconditions:**

The user has selected a specific type of attack/defense.

The selection is saved within the system.

**Main Success Scenario:**

1. The user views a dropdown list labeled 'Select Attack/Defense Type'
2. The user reviews the options, which include various attacks, defenses.
3. The user selects a preferred attack/defense type from the dropdown list.
4. The user confirms the selection by clicking the 'Save' button.
5. The system validates the selection.
6. The chosen type is saved within the system, associated with the user's current model configuration.

**Extensions:**

2a. The user requests more information about each attack/defense type.

2a1. The system provides a brief description and potential impact of each option when hovered over or selected.

**Alternative Flows:**

5a. The system encounters an unexpected error while saving the user's selection.

5a1. The system displays an error message: "Unable to save your selection at this time. Please try again."

5a2. The user is prompted to retry the save operation.

**UC3: Model Evaluation and Reporting**

**Preconditions:**

A model file is successfully uploaded and stored within the system.

User preferences for attack and defense configurations are set and saved within the system.

**Postconditions:**

A comprehensive evaluation report, including visualizations, is generated and presented to the user, detailing the model's performance metrics before and after applying selected attacks and defenses.

**Main Success Scenario:**

1. The system initiates the evaluation process using the stored model file and user-defined attack/defense settings.
2. conduct the evaluations and generate a .json file containing the results.
3. The system parses the .json file to extract relevant evaluation metrics
4. The user navigates to the 'Evaluation Results' section within the system.
5. The system presents a detailed report to the user, which include:  
   Baseline metrics,Post-attack metrics, Post-defense metrics  
   Recommendations for improved defenses, if applicable, based on the evaluation results.

**Extensions:**

5a. The user requests a more detailed breakdown of the results for a specific class or metric.

5a1. The system provides an expanded view with additional data and visualizations for the selected class or metric.

**Alternative Flows:**

2a. The .json file generated by ART contains errors or is incompatible.

2a1. The system displays an error message: "Error processing evaluation results. Please check the evaluation settings and try again."

2a2. The user is prompted to review and adjust the attack/defense configurations before reinitiating the evaluation.

4a. The system encounters an unexpected error while generating or displaying the report.

4a1. The system displays an error message: "Unable to generate the report at this time. Please try again later."

4a2. The user is advised to retry accessing the report after some time or contact support if the issue persists.

5a. ART library issues interfere with the evaluation process.

5a1. The system detects the specific ART issue (e.g., library conflict, outdated ART version).

## **דרישות פונקציונאליות**

|  |  |  |  |
| --- | --- | --- | --- |
| מס''ד | קטגוריה | תיאור | אחראי |
| 1.1 | ניהול מודלים | המערכת חייבת לאפשר למשתמשים להעלות קבצי מודל למידת מכונה, כולל פרמטרים וארכיטקטורה. | יועד |
| 1.2 | ניהול מודלים | המערכת חייבת לאמת את קבצי הדגם שהועלו עבור תאימות ופורמט. | יועד |
| 2.1 | תצורת התקפה והגנה | על המשתמשים להיות מסוגלים לבחור ולהגדיר פרמטרים של התקפה ואסטרטגיות הגנה. | יועד |
| 2.2 | תצורת התקפה והגנה | המערכת חייבת לשמור העדפות משתמש לתצורות התקפה והגנה. | יועד |
| 3.1 | הערכה וניתוח | המערכת חייבת לבצע הערכות של דגמים נבחרים בתנאים רגילים, תחת התקפה ועם הגנות מיושמות. | רון |
| 3.2 | הערכה וניתוח | המערכת חייבת ליצור מדדי הערכה(accuracy,recall וכו') עבור כל תרחיש. | רון |
| 3.3 | הערכה וניתוח | על המערכת לצבור תוצאות הערכה ולספק דירוג של מודלים והגנות בשיטת TOPSIS. | רון |
| 4.1 | דיווח | על המערכת להפיק דוח מקיף המסכם את תוצאות ההערכה, כולל הדמיות ומטריקות. | חן |
| 4.2 | דיווח | הדוח חייב לכלול המלצות למודל והגנות מיטביות המבוססות על ההערכה. | חן |
| 4.3 | דיווח | המערכת חייבת לאפשר למשתמשים להציג, להוריד ולשתף דוח הערכה. | חן |
| 5.1 | ממשק משתמש | המערכת חייבת לספק ממשק משתמש גרפי אינטואיטיבי (GUI) עבור כל אינטראקציות המשתמש, כולל העלאת מודל, בחירת התקפה/הגנה וצפייה בדוחות. | יועד |
| 5.2 | ממשק משתמש | המערכת חייבת לטפל בשגיאות, לספק משוב אינפורמטיבי למשתמשים לפעולות מתקנות. | יועד |

## **דרישות לא פונקציונאליות**

|  |  |  |
| --- | --- | --- |
| מס''ד | קטגוריה | תיאור |
| 1.1 | ביצועים | על המערכת להגיב לקלט משתמשים בזמן סביר. |
| 1.2 | ביצועים | על המערכת לטפל באופן יעיל ובזמן סביר את הערכת המודלים. |
| 2.1 | סקלביליות | על המערכת לתמוך בהוספת התקפות/הגנות חדשות ללא פגיעה ביעילות והערכה. |
| 3.1 | אבטחה | על המערכת להיות מאובטחת ולשמור על מידע רגיש. |
| 4.1 | שימושיות | ממשק המשתמש צריך להיות אינטואיבי עם מינימום זמן הדרכה למשתמש. |
| 5.1 | גיבוי ושחזור | על המערכת לספק מכניזם ליצירת recovery point לשחזור  במקרה של איבוד מידע עד 4 שעות. |
| 5.2 | גיבוי ושחזור | על המערכת לבצע גיבוי כל 24 שעות לכלל המודלים, קונפיגורציות ודו''וחות. |
| 6.1 | תחזוקה | הקוד של המערכת חייב להיות מתועד היטב ולפעול לפי נוהלי הקידוד הטובים ביותר כדי להקל על תחזוקה ועדכונים עתידיים. |

## **תכנון המנשק**

## 

## **שימוש בנתונים**

על מנת לבחון את התוצר שנפתח, אנו נשתמש במודלים מסוג עצי החלטה. את המודלים ואת הנתונים שבהם נשתמש על מנת לבחון את המודל ניקח מכמה מקומות. חלקם קיימים כבר ב-ART, אך על מנת למקסם אץ הדיוק הניתוח ולגוון את הנתונים שהמודל ישתמש בהם, ניקח datasets מספריית pytorch.datasets ונבקש גם מהמשתמש במידה ויש לו. אנו נשתמש בנתונים תחילה על מנת לבחון את תוצאות המודל לפני התקפה, ולאחר מכן נבדוק באמצעותן כיצד ההתקפות השונות משפיעות על תוצאות המודל. לאחר מכן, נבחן כיצד משפיעות ההגנות השונות על תוצאות המודל והאם שימוש בהגנה מסוימת כנגד התקפה מסוימת יכולה להקטין את השפעת ההתקפה על התוצאות שהמודל מחזיר.

## **תוצרים מתוכננים**

## התוצר אותו אנו מתכננים לפתח הוא מערכת אוטומטית לניתוח והערכה של מודלי עצי החלטה שונים. המערכת תקבל מודל וקונפיגורציה עבור ניתוח המודל ותחזיר דוח. בדוח שהמערכת תחזיר, יהיו מפורטים נקודות החוזקה והחולשה של המודל. כלומר, אילו התקפות הצליחו לפגוע יותר במודל ואילו פחות. בנוסף, יופיע בדוח הצעות להגנות שונות על מנת לחזק את עמידות המודל, לאחר שנבחנו אל מול ההתקפות השונות שפגעו במודל.

## **אתגרים וסיכונים**

## אינטרקציה עם ART:

אחד האתגרים המרכזיים שלנו בפרויקט זה הוא ללמוד כיצד ART בנוי, אילו מודלים יש לו ובמה הוא משתמש. בנוסף, עלינו ללמוד כיצד ART פועל, מה תהליך הריצה שלו ומה הוא התוצאות שהוא מחזיר.

## אוטומציה של התקפות והגנות ביעילות:

כאשר ניצור אוטומציות של התקפות והגנות, אנו נבחר הגנה כלשהי למודל, נלביש אותה על המודל שקיבלנו מהמשתמש ונתקיף את המודל. עלינו לבחון את הקומבינציות של ההגנות וההתקפות והפעלתן בצורה שתהיה יעילה ככל שאפשר, על מנת להקטין את זמן הריצה של ניתוח המודל והחזרת התוצאות.

## הערכת חוסן לא מדויקת:

כאשר אנו בודקים את עמידות המודל מול התקפות שונות ובעזרת הגנות שונות, אנו נרצה לחשב את רמת העמידות של המודל בצורה מדויקת ככל שניתן, על מנת שנוכל להחזיר ניתוח מדויק ואמין. במידה ונעריך את המודל הערכה שגויה, הדבר יפדע בבחירת הפתרונות לחיזוק המודל ולבסוף להחזיר מידע שגוי למשתמש.

## פתרונות לא אופטימליים:

לאחר שחישבנו את רמת החוסן של המודל אל מול מתקפות שונות, נרצה להציע פתרונות על מנת להעלות את רמת החוסן של המודל, ללא פגיעה במודל עצמו. הגנות שונות עלולות להוריד מרמת הדיוק של המודל ואנו מעוניינים להציפע פתרון אופטימלי הממקסם את עמידות המודל תוך ניסיון להקטין ככל שניתן את הפגיעה במודל עצמו.

## זיהוי והערכת מודלים לפי קטגוריות:

על מודלים שונים אנו נפעיל התקפות מסוגים שונים. על מנת שההתקפות יתאימו לסוג המודל, אנו נרצה לזהות את סוג המודל נכונה על מנת לבחון עליו התקפות המתאימות לו. לדוגמה, אנו נרצה להפעיל התקפות מסוג whitebox על מודלים מסוג whitebox, והתקפות מסוג blackbox על מודלים מסוג blackbox. במידה ונפעיל התקפות שונות על מודלים מהסוג הלא נכון, אנו עלולים לקבל תוצאות שאינן נכונות ובכך להעריך את המודל הערכה שגויה.

## **כלים ושיטות**

אנו נשתמש בחבילת ART, חבילת קוד פתוח המספקת מספר התקפות, הגנות ודירוגים. אנו ניקח את המודל הקיים ב-ART, נשתמש בכלים שלו עבור הפיתוח והאוטומציות. לאחר שהמערכת תעריך את המודלים באופן אוטומטי, אנו נפתח שיטות נוספות על מנת להעריך את המודל (עוד התקפות, הגנות ודירוגים שונים) אותן נוסיף למערכת על מנת להשיג תוצאות מדויקות יותר עבור הערכת המודלים.

## **לו''ז מעודכן לפרוייקט**

## תמונה שמכילה צילום מסך, טקסט, צבעוני, תרשים התיאור נוצר באופן אוטומטי**חלוקת אחראיות**

רון – פיתוח הגנות ודירוגים

חן – פיתוח התקפות, דירוגים ואוטומציות

יועד – אחראי GUI ו-testing

# **ביבליוגרפיה**

Hwang, C.-L., & Yoon, K. (1981). Technique for order of preference by similarity to ideal solution (TOPSIS). Journal of Decision Systems, 1, 123-160.