Iby and Aladar Fleischman
Faculty of Engineering
Tel Aviv University

הפקולטה להנדסה
ע"ש איבי ואלדר פליישמן
אוניברסיטת תל-אביב

דו"ח מסכם פרויקט בקורס מבוא ללמידת מכונה

מדעים דיגיטליים להייטק

מגישות:

שלי פייגין

שני חזן

יוני 2023

אודות הפרויקט

הנתונים שלנו מכילים מידע אודות קבצי הרצה (.exe). במסגרת הפרויקט הובאה לידינו בעיית סיווג בינארי (Classification Binary), בה נדרשנו לסווג רשומות לשתי קטגוריות: האם הקובץ זדוני (malicious) או לא (benign) באמצעות ניתוח סטטיסטי של הנתונים –ניתוח המידע אודות הקבצים מבלי להריץ אותם.

במהלך הפרויקט נעשתה עבודת חקר על הנתונים והפיצ'רים שחלקם ידועים וחלקם אנונימיים על מנת לעמוד על טיבם בהקשר לחיזוי זדוניות הקבצים. הנתונים עובדו בעזרת שיטות מגוונות והותאמו על מנת שהמודלים יוכלו ללמוד מהם. לאחר ניסויי מודלים רבים, נמצא כי מודל **Random Forest** סיפק את הביצועים הטובים ביותר ובאופן עקבי למען מטרה זו. המודל מפיק חיזוי בגובה של **0.97** במדד AUC בעזרת שיטת **5-Fold Cross-Validation**.

חילקנו את העבודה על הפרויקט לחמישה חלקים עיקריים: תחילה ביצענו אקספלורציה והפעלנו ויזואליזציות על סט הנתונים 'train' וקיבלנו מושג אודות הפיצ'רים. לאחר מכן, עברנו לשלב העיבוד המקדים בו פעלנו לפי מסקנות השלב הקודם וביצענו שינויים בנתונים כגון: הסרת פיצ'רים שאינם תורמים ללמידה, מניפולציה מתמטית על פיצ'רים קיימים, התמודדות עם פיצ'רים קטגוריאליים, טיפול בערכים חסרים, נרמול הנתונים והורדת ממדים. בשלב השלישי והרביעי הרצנו מודלים שונים שלמדנו במסגרת הקורס והערכנו את ביצועיהם ובחרנו את המודל האופטימלי עבורנו. לבסוף בשלב החמישי ביצענו תחזית על סט ה'test' שקיבלנו. כעת נפרט על כל אחד מהם.

חלק ראשון – אקספלורציה

תחילה ייבאנו את סט ה-'train' ובחנו את כמות המידע ועל אילו ממדים הוא מתפרש. גילינו שבסט הנתונים ישנן 24 עמודות, כלומר 23 features ו- 'label'. בנוסף, סט הנתונים מכיל כמות יפה של 60,000 דוגמאות.
כבר בשלב זה הבחנו שהפיצ'ר '**sha256**' הינו מזהה ייחודי של קובץ (ID) ולכן החלטנו להסירו בהמשך בהנחה שאינו תורם למודל.
<u>ראינו שסוגי הפיצ'רים נחלקים לשניים</u>: מספריים (המתפרשים על טווחי מספרים שונים) וקטגוריאליים (בעלי כמות משתנה של קטגוריות שהינן מחרוזות), בחנו את סוגי הערכים השונים והכמות של כל אחד מהם והנחנו כי בהמשך נצטרך להתמודד עם הפיצ'רים הקטגוריאליים 'C', 'file_type_trid' מכיוון שחלק מהמודלים שבהם נשתמש בהמשך ידרשו רק ערכים מספריים כדי לפעול.
בדקנו כמה קבצים בסט הנתונים מסווגים כ- '1' (זדוניים) וכמה מסווגים כ- '0' (שפירים) וגילינו שהחלוקה **מאוזנת** – חצי חצי בדיוק, הדבר מאפשר לנו חלוקה מאוזנת לסט 'train' ולסט 'validation' בהמשך (נספח 1).
בשלב הבא, בחנו כמה ערכים חסרים יש בכל עמודה באחוזים וגילינו שמלבד 'label', ל4 פיצ'רים נוספים אין ערכים חסרים כלל ולשאר כמות מועטה של ערכים חסרים אותם נשלים. (נספח 2)
הצגנו את ההתפלגויות הפיצ'רים המספריים בכדי לבדוק כיצד הם מתפלגים (נורמלית, הטיה ימינה, שמאלה וכו'). לאחר התבוננות בגרפים גילינו כי **ישנם 5 פיצ'רים מתוכם שהם בינאריים (ערכים 0/1)** כך שאין משמעות להתפלגות שלהם (נספח 3), לכן הצגנו פעם נוספת את ההתפלגויות בלעדיהם אך הפעם עם **'Log'** בכדי לבדוק האם הוא מציג אותן בצורה נקייה יותר והתוצאה הייתה חיובית אז החלטנו להחיל בהמשך את 'Log' על כל הפיצ'רים המספריים הרציפים (נספח 4).
על סמך התוצאות, סיווגנו את הפיצ'רים המספריים הרציפים ל-<u>3 התפלגויות נפוצות</u>: 5 פיצ'רים שהתפלגותם קרובה לנורמלית, 7 בעלי התפלגות מוטה ימינה (מוטה חיובי), אחד שמוטה שמאלה (מוטה שלילי) ושניים שהתפלגותם לא מוכרת, ניקח זאת בחשבון כשנטפל בערכים החסרים.
בשלב הבא הצגנו מטריצת קורלציות בכדי להבין אם קיים קשר בין הפיצ'רים השונים (נספח 5). חקרנו את שני הזוגות שהיו בעלי קורלציה גבוהה במיוחד וגילינו שלפיצ'ר 'size' קיימת קורלציה גבוהה גם עם הפיצ'ר

'num_strings' (0.91) וגם עם הפיצ'ר 'MZ'(0.72), לכן החלטנו שנרצה להסיר את פיצ'ר '**size**' בהמשך כי הוא כבר מוסבר בצורה מספיק טובה על ידי פיצ'רים רבים כולל שניהם. ניתן לראות שתי ויזואליזציות המתארות את הקשר הכמעט לינארי של 'size' עם כל אחד מהפיצ'רים הנ"ל (נספח 6).
לאחר מכן, בחנו האם קיים קשר בין זדוניות הקובץ 'label' לכל אחד מחמשת הפיצ'רים הבינאריים גילינו שלפיצ'ר **'has_relocations'** יש כמות יחסית זהה של קבצים זדוניים ושפירים, הדבר מרמז לנו שלפיצ'ר זה אין כמעט משמעות בסיווג הקובץ כזדוני\ שפיר ולכן יוסר (נספח 7). לסיום, בחנו את שני הפיצ'רים הקטגוריאליים שלנו (מלבד 'sha256' שבכוונתנו להסיר) 'C' ו'file_type_trid'. בדקנו בכל אחד מהם באילו קטגוריות יש יותר קבצים זדוניים וגילינו **בפיצ'ר 'file_type_trid' 4 קטגוריות (סוגי קבצים) שכל הקבצים בהן תמיד זדוניים!** בפיצ'ר 'C' גילינו שהקטגוריה שכוללת הכי הרבה קבצים זדוניים מבין הקטגוריות האחרות היא 'vh' (נספח 8).

חלק שני – עיבוד מקדים

הפעולות בשלב העיבוד המקדים בוססו על סמך מסקנותינו מהשלב הקודם וחולקו לשישה חלקים, נדגיש כי העיבוד המקדים על סט הנתונים 'test' ייעשה מיד לאחר סיום העיבוד המקדים על נתוני 'train'.
לפני החלת שינויים בסט ה- 'train', החלטנו להפריד 10% מתוכם שיהיו 'validation' ולהעניק להם את ה-"test treatment", כלומר לבצע עליהם את כל העיבודים שנבצע בהמשך על נתוני ה-'test', ולא לאפשר לאף מודל ללמוד מהם באופן ספציפי. דבר זה אפשר לנו להעריך ולוודא את המודל בצורה יותר מדויקת (validation), שכן לנתוני ה -'test' הסופי אין לייבלים ולא נוכל לדעת כמה המודל יחזה במדויק עבורם. ההחלטה לבחור עשירית מהנתונים באה מתוך שיקול של איזון בין כמות נתונים מספקת ללמידה עבור המודלים, אל מול בחירת כמות תצפיות שתספק לנו תוצאות אמינות ומשכנעות. <u>נציין כי הדבר היחיד שנימנע ממנו הוא להסיר ערכים חריגים מסט ה'test' וה-'validation'</u>.

1. <u>הסרת פיצ'רים לא רלוונטיים</u>- בחלק הקודם החלטנו להסיר 3 פיצ'רים שהנחנו שלא יתרמו לנו למודל ולסיווג הנתונים:
   - 'sha256' – משמש כאות זיהוי ייחודי לכל קובץ ואינו קשור לאיתור זדוניות הקבצים.
   - 'has_relocations' – כמות יחסית זהה של קבצים זדוניים ושפירים, אזי אין לו תרומה לסיווג הקובץ.
   - 'size' – בעל קורלציה גבוהה עם משתנים אחרים שמסבירים אותו בצורה מספיק טובה, לכן ניתן לוותר עליו.

2. <u>מניפולציה מתמטית על פיצ'רים קיימים</u>- כפי שראינו באקספלורציה, פונקציית 'log' מציגה את התפלגויות הפיצ'רים בצורה נקייה וברורה יותר, לכן שינינו את כל הפיצ'רים המספריים הרציפים (ללא הבינאריים שאין להתפלגותם משמעות) לגרסת ה-log שלהם (החלנו עליהם את פונקציית log).

3. <u>טיפול בערכים חסרים וההתמודדות עם הפיצ'רים הקטגוריאליים</u>- הגישה בחלק זה התבססה על השלמת ערכים באופן הגיוני ואחיד אשר מתואם עם הנתונים שכן קיימים, כך שלמעשה ננסה להיצמד כמה שיותר למה שהנתונים מספרים לנו. יישום של גישה זו בא לידי ביטוי בכך שביצענו זאת בנפרד עבור הפיצ'רים הקטגוריאליים, הבינאריים והמספריים הרציפים.
   - **הפיצ'רים הקטגוריאליים:** לפני הטיפול בערכים החסרים רצינו להמירם למספריים. מלכתחילה, לא מצאנו סיבה להפוך כל קטגוריה בכל אחד מהפיצ'רים לעמודה חדשה כי הפשטות הזו הייתה עולה לנו בהגדלת ממדיות הבעיה בצורה משמעותית בתוספת של 94 עמודות חדשות (94=88+6) ולכן החלטנו לתת לכל קטגוריה מספר סידורי מ1 והלאה כך שהקטגוריה שקיבלה את המספר הסידורי הכי קטן היא הנפוצה ביותר מבין הקטגוריות האחרות באותה העמודה וכן הלאה. כך, נשארנו עם ממד בגודל סביר, כאשר בהמשך שיטות נוספות להפחתת ממד אף שיפרו את הביצועים בצורה משמעותית. באופן שרירותי השלמנו את הערכים החסרים בכל עמודה בערך ה-

חציון בהנחה שאיזון הערכים יישמר. בהמשך ניסינו להשלים גם ע"י ממוצע אך הביצועים ירדו ולכן נשארנו עם החציון. בכל זאת, נתקלנו בבעיה עם הפיצ'ר 'file_type_trid' בו הקטגוריות מוספרו מ-'1' עד '89' (במספר הקטגוריות). ראינו שכאשר אנו ממלאים את הערכים החסרים בסט ה-'test' קיימות קטגוריות נוספות שלא היו קיימות לנו בסט ה- 'train'. טיפלנו בבעיה בכך שהשלמנו את הערכים החסרים בהן ב-'0' כדי לייצר הפרדה בין המספרים שכבר החלנו על שאר הקטגוריות.

- **הפיצ'רים הבינאריים (0/1):** בויזואליזציות של הפיצ'רים הבינאריים בחלק הקודם ראינו כי בכולם הערך '1' גבוה משמעותית מהערך '0' ולכן השלמנו בערכים החסרים של כל עמודה את הערך '1' על מנת לשמור על יחס זה שכן הסיכוי לקבל '1' גבוהה יותר מלקבל '0'.
- **הפיצ'רים המספריים רציפים: לצורך כך יישמנו את מודל ה- KNNImputer שלא למדנו עליו בקורס.** המודל משלים ערכים חסרים לפי **ממוצע ערכי K השכנים שלו** ובכך משפיע על דיוק הערכים- יעלה בהמשך את ציוני המודלים. בחרנו את K להיות שורש ריבועי של מספר הדוגמאות כיוון שקראנו שזה ערך ברירת המחדל המומלץ ביותר.

4. <u>בעיית גודל הממדיות</u> - ממדיות גדולה מדי של בעיה עלולה להוליך אותנו שולל לכדי Overfitting משום שהמודל יותאם יתר על המידה לסט הנתונים 'train' ונשלם על כך בשונות גבוהה מאוד ולהטיה נמוכה בביצועי המודל על סט ה-'test'. החוכמה והגדולה היא למצוא את האיזון הנכון שכן המידע מצוי בפיצ'רים עצמם שמספקים לנו את הביצועים המעולים אך שימוש יתר בהם יגרור פיחות משמעותי בביצועים בהמשך. ממד הבעיה שלנו תלוי <u>במורכבות המודל</u>. עבור מודל מורכב איננו רוצים להפחית את הממדיות כי זה רק פוגע בזמן הריצה וסיכון להתאמת יתר. לעומת זאת, עבור מודל פשוט עלולה להיווצר בעיית התאמת יתר ולכן נרצה להקטין שם את הממדיות. למשל במודל Logistic Regression (מורכבות נמוכה) אנחנו כן מורידים את הממדים, אולם במודל Random Forest (מורכבות גבוהה) ממד הבעיה שלנו לא משתנה ולכן אנחנו לא מורידים שם את הממדים.
נסביר כבר מעכשיו שבעבור כל מודל ניסינו מספר פיצ'רים שונה וכך ניסינו להגיע למספר אופטימלי שיענה על ה Bias-Variance Tradeoff ולקחנו את המספר שנתן לנו את התוצאות הטובות ביותר.
למעשה, לאחר שכבר הסרנו 3 פיצ'רים שאינם רלוונטיים, הרצנו 2 שיטות שלמדנו בקורס: שיטת הPCA ושיטת ה- Forward Selection על המודלים **הבסיסיים** שבחרנו וביצענו השוואה בין 2 השיטות, זו שנבחרה הייתה בעלת שגיאה נמוכה יותר. השוואה זו שיפרה את הביצועים משמעותית.

5. <u>איתור וטיפול בערכים חריגים</u> – נתונים חריגים עלולים להוות מכשול למודל כאשר הלמידה מתבצעת על נתונים שלא סביר שנקבל בעתיד (לכן חריגים) ואם נלמד עליהם ייתכן והם ייטו את התחזיות. אנו רוצים לנקות או לטפל בהם טרם הכנסת הנתונים למודל כדי להימנע מבעיה זו.
נתבונן בערכים הקיצוניים של הפיצ'רים המספריים הרציפים בלבד כי אין משמעות לערכים קיצוניים בפיצ'רים בינאריים וקטגוריאליים (נספח 9). מהתבוננות ב- boxplots גילינו שישנם שני פיצ'רים להם אין ערכי קיצון ולכן התעלמנו מהם. בשאר הפיצ'רים המספריים הרציפים ראינו שיש הבדלים במיקומי הימצאות הערכים הקיצוניים מעל הגבול העליון/מתחת לגבול התחתון/בשניהם, לכן טיפלנו בכל אחד מהם בנפרד.
הערכים החריגים יחתכו בשיטת הטווח הבין-רבעוני (IQR) הכוללת מחיקת ערכים בעלי ערך מעל האחוזון ה-95% ומתחת לאחוזון ה-5% . לא רצינו לטפל ביותר מדי ערכים כאלו כי אנו חושדים שלשונות יש חשיבות והיות שלסט ה-'test' לא מבוצעת הורדת ערכי קיצון. לאורך כל העבודה עשינו הלוך ושוב ושינינו את בחירת האחוזון שיחתוך את ערכי הקיצון וגילינו כי הנ"ל מניבים את התוצאות הטובות ביותר. ניכר מהתוצאות שצפיפות הערכים גדלה והטווח ירד- משמע הנקודות קרובות יותר.
ההנחה החשובה שהעלנו מכאן היא שעקב כך שהפיצ'ר **'symbols'** הכיל כמות גדולה מאוד של ערכים חריגים ואחרי החיתוך כל ערכיו שווים ל-0, כנראה שהפיצ'ר בקושי יתרום למודל. אכן, בהמשך הפעלנו על המודל Random Forest את ה-feature importance וההנחה התגלתה כנכונה.

לסיכום, דיווחנו על מספר הערכים שהשתנו על מנת שתהיה לנו בקרה על כך: הסרנו 4,820 דוגמאות מתוך 60,000 הדוגמאות בסט הנתונים שלנו שמהווים 8%- כמות סבירה לגמרי. (נספח 10).

6. <u>נרמול סט הנתונים</u>- הנרמול אמור להיות חשוב בעבור שיטות הורדת ממדים כמו PCA ובעבור מרבית המודלים (פרט לעצים למשל) משום שכל פיצ'ר נתון בסקאלה שונה כך שהיחסיות בניהם הופכת חשובה כאשר פיצ'ר מסוים יוכל לקבל משקל גדול יותר לא מסיבה מוצדקת אלא רק מפני שהסקאלה שלו בערך אבסולוטי הרבה יותר גבוהה. בחלק הקודם ראינו את התפלגויות הפיצ'רים שכן חלקם דומים להתפלגות נורמלית וחלקם לא, לכן ניסינו 2 שיטות במהלך העבודה: MinMaxScaler, ו-StandardScaler, השיטה שעבדה לנו טוב יותר היא: StandardScaler שמנרמלת את הנתונים והופכת את התוחלת של כל פיצ'ר ל-0 והשונות ל-1.

חלק שלישי ורביעי – הרצת המודלים והערכתם

בכל המודלים השתמשנו במתודולוגיה דומה כאשר אנו בוחנים מספר היפר-פרמטרים בעבור כל מודל ולאחר מכן ה-GridSearchCV בוחן שוב את הקומבינציה האופטימלית של ההיפר-פרמטרים עם הפיצ'רים הנבחרים. את המודלים הערכנו בעזרת Cross Validation אשר מספק דיוק רב יותר של הערכת המודל על גבי חלוקות שונות של train and validation וממצע ביניהן. נוסף על כך נעזרנו בגרף שמצייר את ציוני ה'ROC AUC' לכל חלוקה שונה כך שיכולנו לאמוד גם את עקביות התוצאות. עבור כל אחד מהמודלים בדקנו את ביצועיו על סט ה- 'validation' שהפרשנו בתחילת העבודה על מנת לדמות מצב מבחן אמת על נתונים שהמודלים לא ראו מעולם וייתן לנו תחושה לגבי ביצועי המודל בעולם שבחוץ. נעזרנו בכך גם על מנת לבדוק overfitting במודלים השונים. ראינו כיצד התוצאות על סט ה-'test' משתנות בהתאם לשינוי פרמטרים שיורידו overfitting ובחרנו את הערך שהתכתב הכי טוב גם עם המלצת GSCV וגם עם בדיקת ה-overfitting הידנית שלנו.

<u>מודלים בסיסיים</u>:

Logistic Regression

ה-GridSearch בחר ממספר היפר-פרמטרים של שיטת רגולריזציה שונה (l1, l2, elasticnet, none) וערכי C שונים שמעידים על רמות הענשה שונות לממדיות גדולה. במודל הסופי לא נבחרה רגולריזציה ו-C=0.001. ראינו כי מספר הפיצ'רים ששיפרו את ביצועינו בצורה המשמעותית ביותר היו 17 ע"י PCA.
התוצאות הסופיות של המודל היו עקביות.
בעזרת גרף 'ROC CURVE' על 5 פיצולי הנתונים ראינו כי הביצועים אחידים מאחר וסטיית התקן נמוכה.

Naïve Bayes Classifier

מודל זה אינו מקבל פרמטרים ולכן הורץ ישירות אך עדיין לא השגנו שיפור ניכר.

<u>מודלים מתקדמים</u>:

Multi-Layer Perceptron (ANN)

כאן יוחסה חשיבות רבה יותר ל-Bias-Variance Tradeoff. מספר שכבות רב מדי יגרור overfitting ולכן הגבלנו את מספרן תוך מתן תשומת לב להתאמת יתר. כמו כן נתנו קצבי למידה שונים ואדפטיביות למידה על מנת לנסות להתכנס למינימום גלובלי ולא להיעצר במינימום מקומי. קבלנו תוצאות עקביות בגובה 0.953 על ה-'train' <u>שמהוות שיפור מהמודלים הבסיסיים</u>. בבדיקת הoverfitting לא עלה ממצא חריג.
(הגענו לציון 0.936 על ה-'validation').

Random Forest

המודל הנבחר. ההיפר-פרמטרים השונים היו מספר העצים ביער, עומק העצים, מינימום דגימות לעלה וקריטריון פיצול. כאשר יער גדול יותר מקטין את השונות אך צורך זמן וכוח חישוב גדול, מנגד, עץ אחד יכול לייצר התאמה לא מידתית לנתונים מסוימים. ככל שהיער גדול יותר כך גם ניתן לאפשר עומק עמוק יותר ולהקטין את השונות עוד יותר. שיחקנו רבות על שילוב פרמטרים אלה בטווח של 100-1000 במספר העצים ובעומק של 5-20 . לאחר הרבה וריאציות תוך התחשבות במסובכות המודל מצאנו שהערך האידיאלי הוא 200 עצים עם עומק 15. רצנו על מספר שונה של מינימום דגימות לעלה ועל שני קריטריונים של פיצול שלמדנו: ג'יני ואנטרופיה. האנטרופיה נבחרה עם מינימום 5 דגימות בעלה. **כאן כבר קיבלנו את הביצועים הטובים ביותר: 0.970 ב-ROC AUC ואף התוצאה על סט ה-'validation' ששמרנו בצד הייתה טובה יותר: 0.972.** התוצאות נראות מרשימות ואנו מרוצים מאיכות המודל.

הידיעה שRandom Forest יודע לעבוד טוב עם נתונים טבלאיים גרמו לנו לבחור במודל זה! (נספח 11)

<u>חשיבות הפיצ'רים במודל</u>: לאחר בחירת המודל הטוב ביותר, בחנו את התרומה של כל פיצ'ר למודל (נספח 12). ניכר כי הפיצ'ר התורם ביותר למודל הוא פיצ'ר 'avlength' שמשמעותו ממוצע אורך המחרוזות בקובץ. אינטואיטיבית, הדבר הגיוני. קבצים זדוניים, לעומת שפירים מכילים בדרך כלל מחרוזות באורכים לא רגילים – קצרות מאוד או ארוכות מאוד שעלולות להכיל כוונות זדוניות. מחרוזות ארוכות מהרגיל עלולות להסוות בתוכן פקודות זדוניות או מסרים מקודדים, בעוד שמחרוזות קצרות מאוד מיועדות לעתים לצמצום "טביעת רגל" של פונקציות זדוניות. הדבר מסביר למה הפיצ'ר כה חשוב למודל שלנו.

<u>Confusion matrix</u>: התקדמנו לבחינת המודל בהקשרי ציוני ה- Accuracy, precision, sensitivity, specificity. לשם כך, הצגנו מטריצות בלבול של סט ה-'train' וסט ה- 'validation' (נספח 13). הסיבה שהצגנו את שניהם היא בכדי לוודא שוב שאנחנו לא נמצאות במצב של overfitting. כלומר, ציפיותינו היו שהמטריצות יהיו די דומות ובעלות הבדלים מינוריים בלבד, וכך הדבר. ציון ה – accuracy של ה-'validation' (אחוז הקבצים המסווגים נכון) הוא 91%. ציון ה- precision שלו (כמות הקבצים שסווגו כזדוניים והם אכן זדוניים) הוא 92%. ציון ה- sensitivity (סיווג נכון של הקבצים כזדוניים מתוך כל הקבצים הזדוניים) שלו הוא 89% ולבסוף, ציון ה- specificity (סיווג נכון של הקבצים כשפירים מתוך כל הקבצים השפירים) על סט הולידיישן הוא 92%. כל הציונים טובים למדי וקרובים יחסית גם לסט ה-'train' כך שאנחנו לא במצב של overfitting.

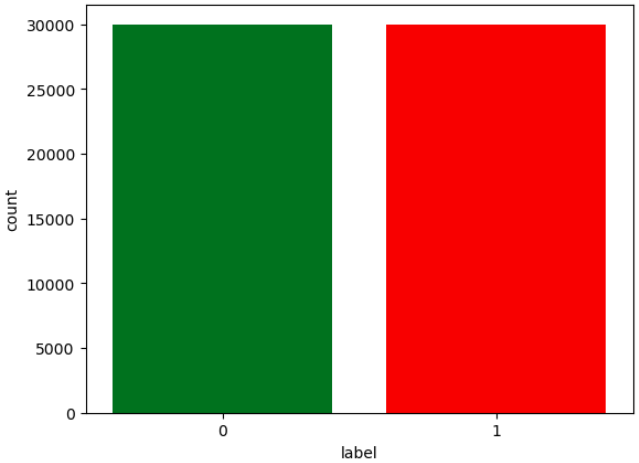
חלק חמישי – ביצוע פרדיקציה

תהליך האקספלורציה והמחקר סיפק לנו תובנות חשובות אשר יושמו במהלך העיבוד המקדים והמידול. בחינה של מודלים רבים בגרסאות שונות הניבה לנו את מודל ה-Random Forest הנבחר עם מספר עצים : 200, עומק עץ: 15, מינימום דגימות בעלה: 5, קריטריון: אנטרופיה. מודל זה חוזה בהסתברות של 0.972 פרדיקציה נכונה בהתבסס על ה-validation שלנו ובהסתברות של 0.97349 פרדיקציה נכונה בהתבסס על סט האימון. המודל נבנה והשתפר בהליך קפדני תוך תשומת לב רב לממדיות סט הנתונים שהוכנס אליו ודגש רב על מנת למצוא את הנקודה האופטימלית עבורנו מבחינת Bias-Variance, תוצאותיו עקביות ואמינות. לבסוף ביצענו תחזית על כל אחת מהתצפיות בסט המבחן ויצאנו את הנתונים לקובץ SDV שבו מתואר מה הסיכוי של כל קובץ להיות זדוני ('1'). הקפדנו בכל שלבי הפרויקט לא ללמוד מה- Test אלא רק להחיל עליו את מסקנות העיבוד על ה- Train וכמובן לא לקצץ מהtest ערכים קיצוניים.

חלוקת העבודה - העבודה נעשתה במשותף, שילבנו מפגשים פרונטליים אך השתמשנו בעיקר בזום שיתופי. את המודלים רצינו לחלק בינינו אך בפועל העבודה התערבבה ושתינו עבדנו על הכל. בפועל, לא הייתה חלוקת אחריות מוסדרת ושתינו עסקנו והיינו מעורבות בכל החלקים כך שכל אחת הביאה את הצד החזק שלה.

נספחים

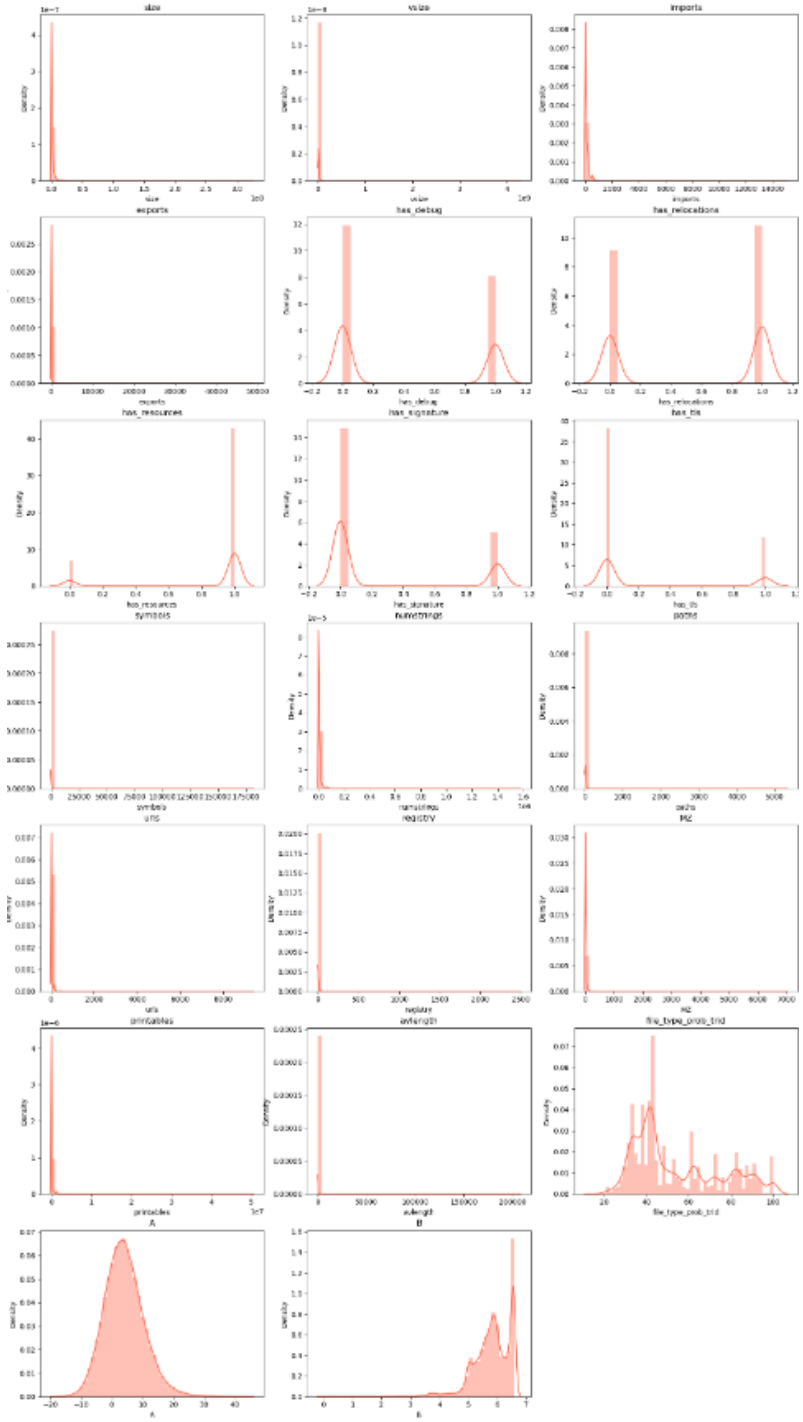
נספח 1- סט הנתונים מאוזן על פי ה-'label'



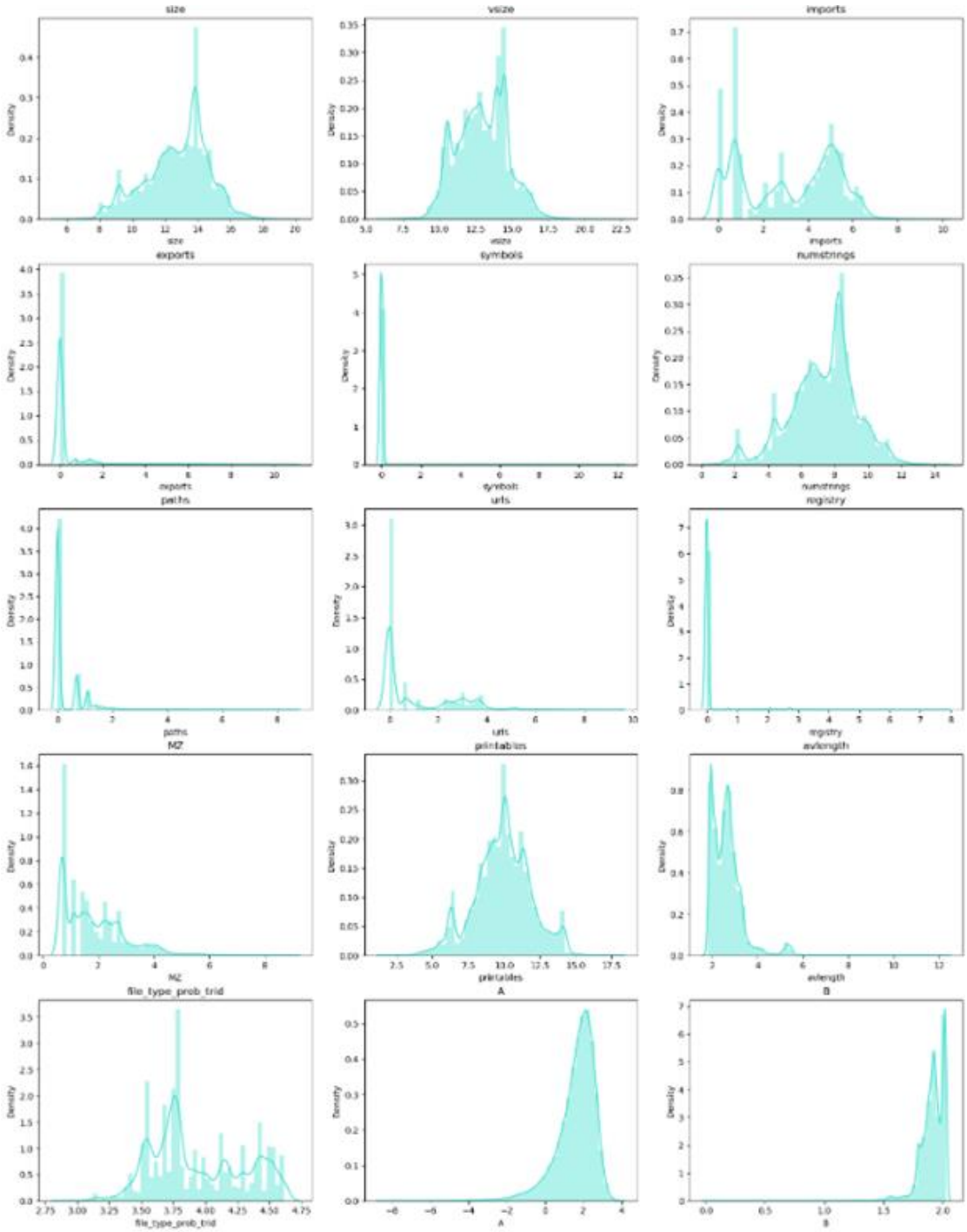
נספח 2- כמות הערכים החסרים בכל פיצ'ר

	Feature	% Null values
0	B	6.25
1	A	6.17
2	paths	6.10
3	has_relocations	5.54
4	MZ	5.15
5	has_debug	4.88
6	has_tls	4.83
7	avlength	4.59
8	printables	4.57
9	numstrings	4.53
10	symbols	4.43
11	registry	4.21
12	urls	3.91
13	exports	3.49
14	C	3.42
15	has_resources	3.27
16	has_signature	3.23
17	vsize	3.23
18	imports	2.90
19	sha256	0.00
20	file_type_prob_trid	0.00
21	file_type_trid	0.00
22	size	0.00
23	label	0.00

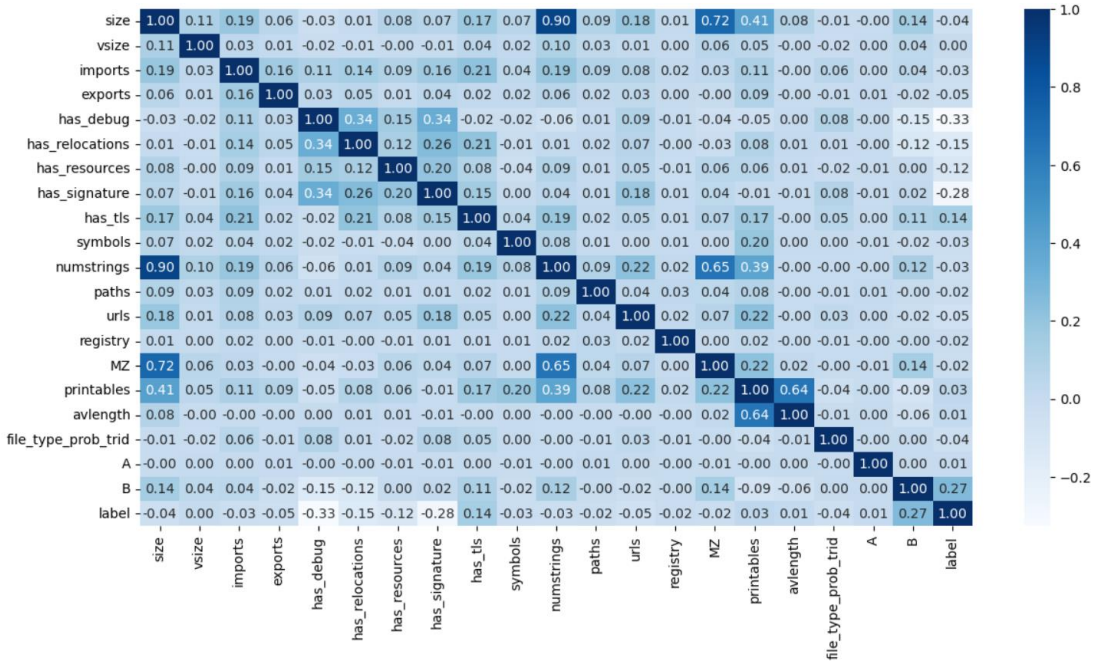
נספח 3 – ההתפלגויות של הפיצ'רים המספריים



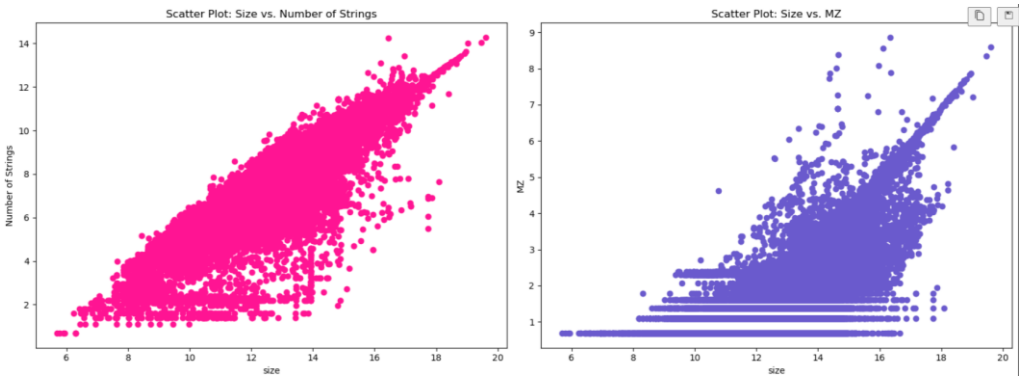
נספח 4 – ההתפלגויות של הפיצ'רים המספריים (בלי פיצ'רים בינאריים) עם 'Log'



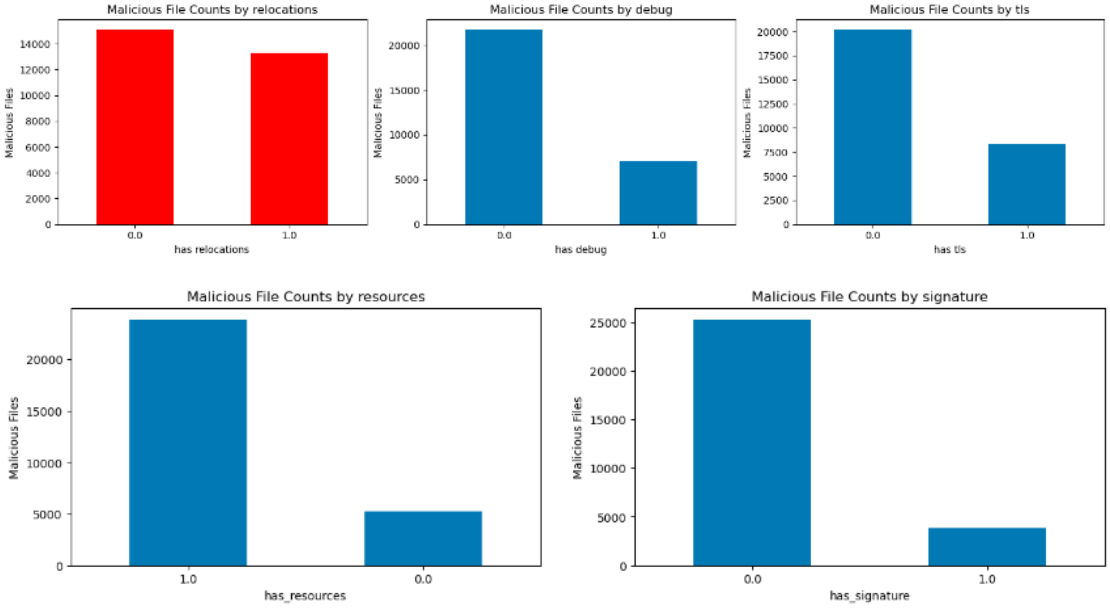
נספח 5- מטריצת קורולציות



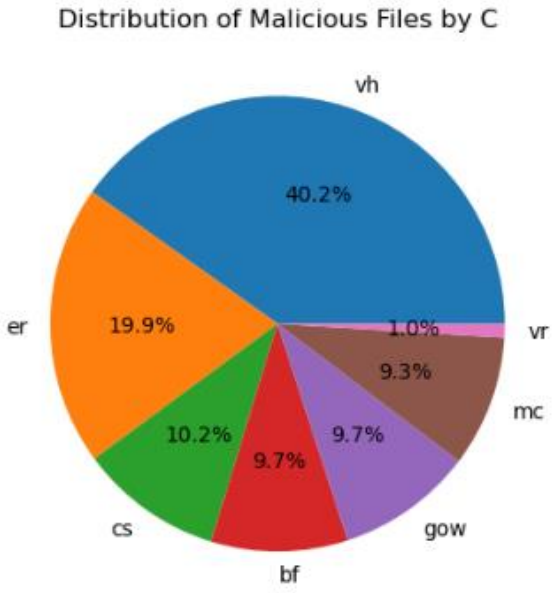
נספח 6 –ע"י scstter plot : קורולציה בין 'size' ל- 'MZ' ובין- 'size' ל- 'num_strings'

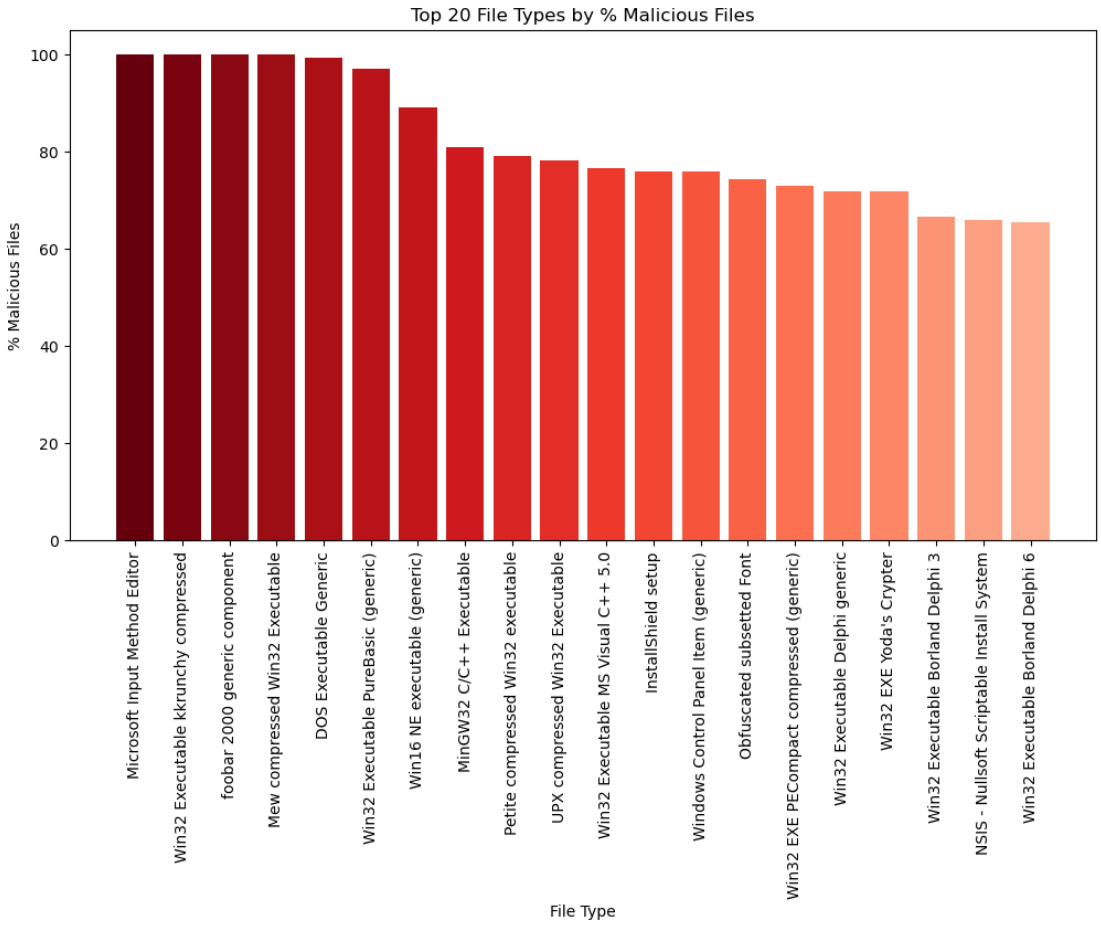


נספח 7 – הקשר בין זדוניות הקובץ 'label' לכל אחד מחמשת הפיצ'רים הבינאריים

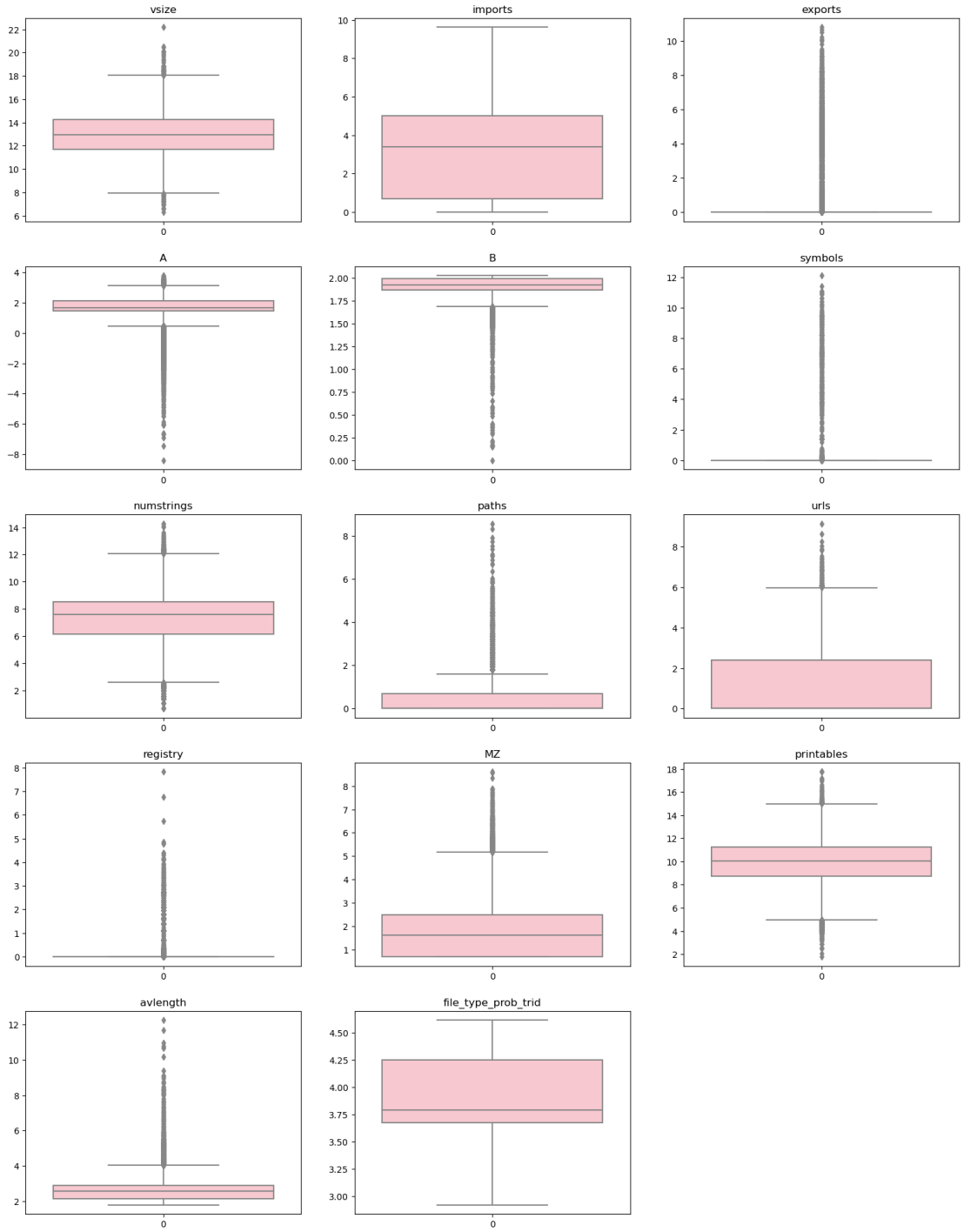


נספח 8 – כמות הקבצים הזדוניים בכל קטגוריה של הפיצ'רים הקטגוריאליים C ו- file_type_trid,

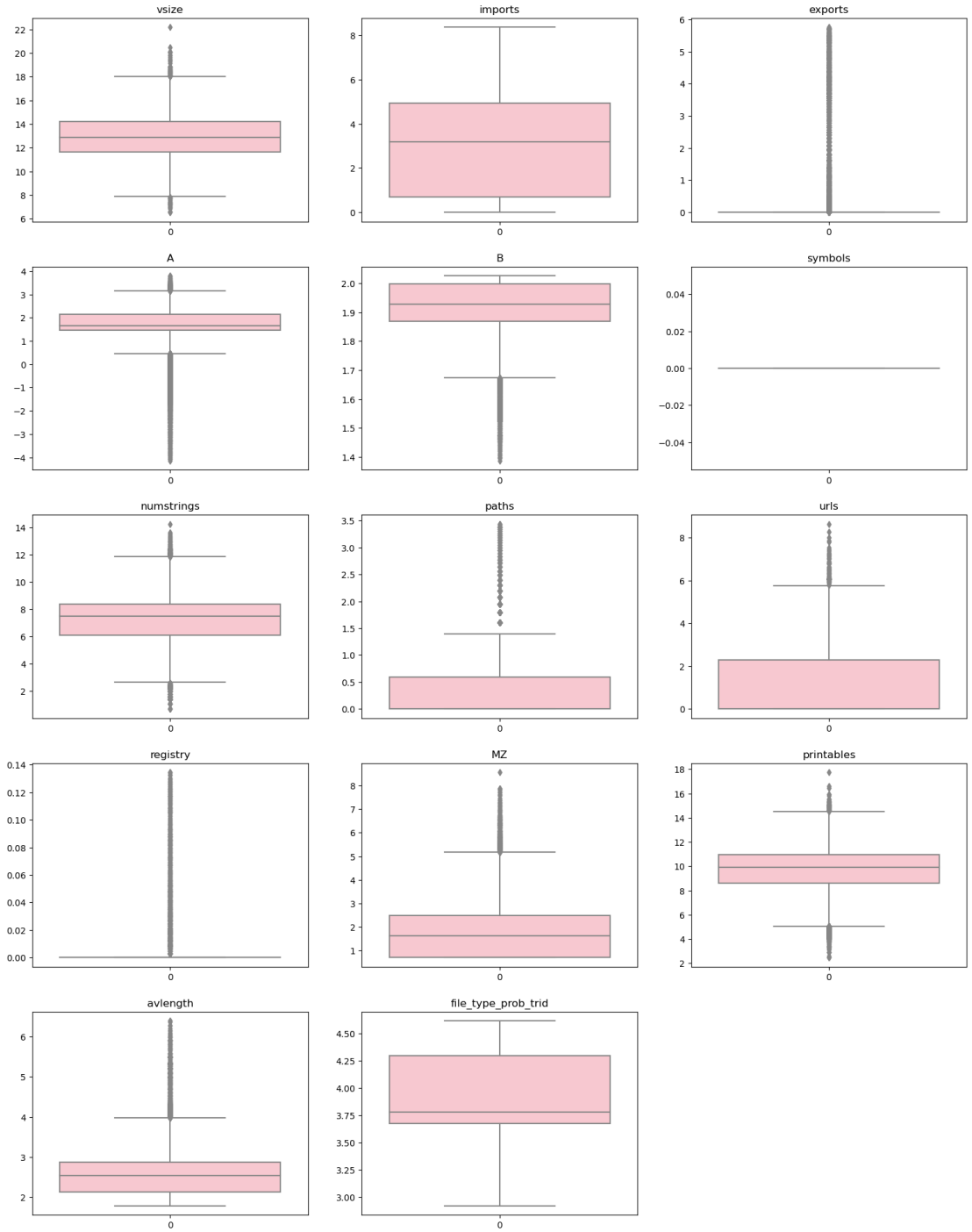



Top 20 File Types by % Malicious Files
% Malicious Files
0
20
40
60
80
100
Microsoft Input Method Editor
Win32 Executable kkrunchy compressed
foobar 2000 generic component
Mew compressed Win32 Executable
DOS Executable Generic
Win32 Executable PureBasic (generic)
Win16 NE executable (generic)
MinGW32 C/C++ Executable
Petite compressed Win32 executable
UPX compressed Win32 Executable
Win32 Executable MS Visual C++ 5.0
InstallShield setup
Windows Control Panel Item (generic)
Obfuscated subsetted Font
Win32 EXE PECompact compressed (generic)
Win32 Executable Delphi generic
Win32 EXE Yoda's Crypter
Win32 Executable Borland Delphi 3
NSIS - Nullsoft Scriptable Install System
Win32 Executable Borland Delphi 6
File Type

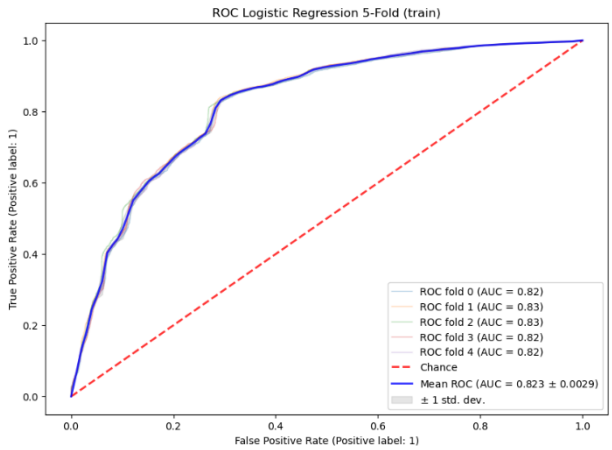
נספח 9 – הצגת Boxplots של הפיצ'רים המספריים הרציפים לפני טיפול בערכים קיצוניים

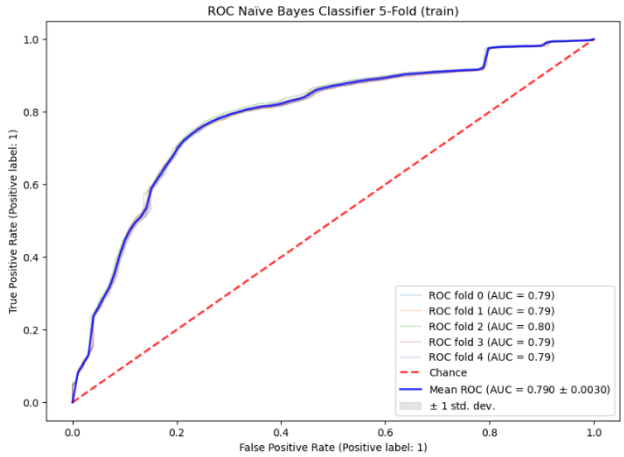


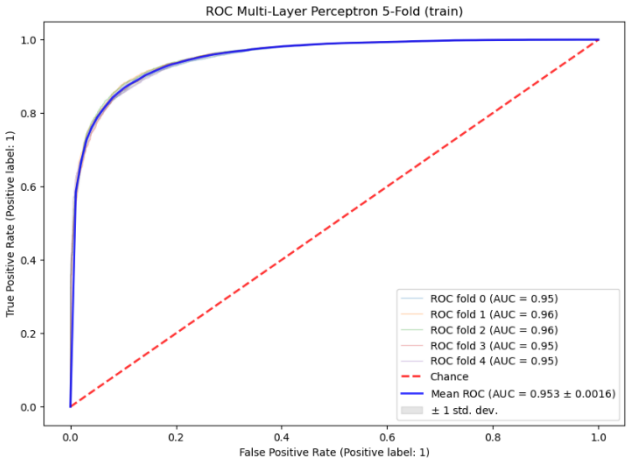
נספח 10 – הצגת Boxplots של הפיצ'רים המספריים הרציפים אחרי טיפול בערכים קיצוניים

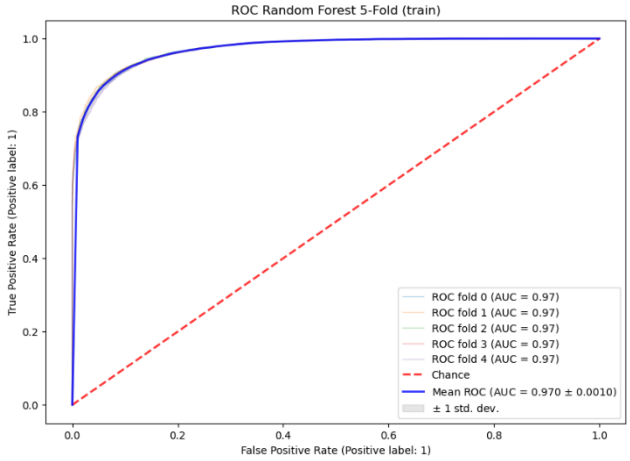


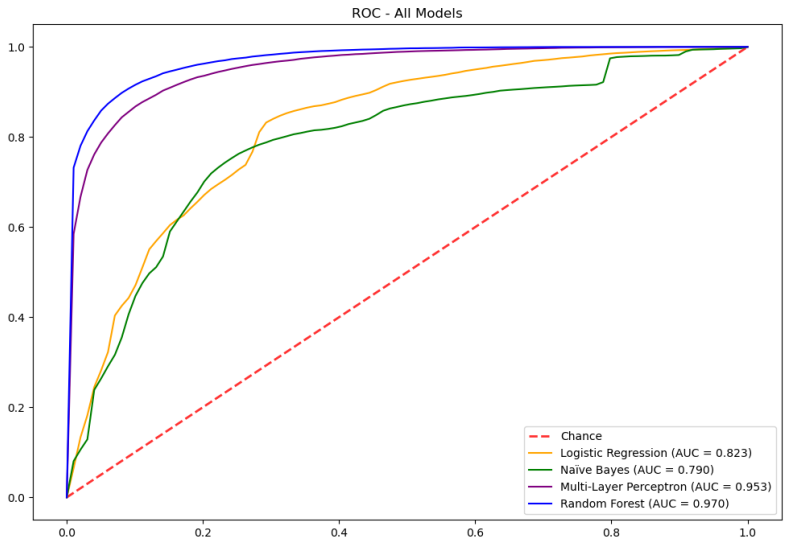
נספח 11- ROC kFold של 4 המודלים



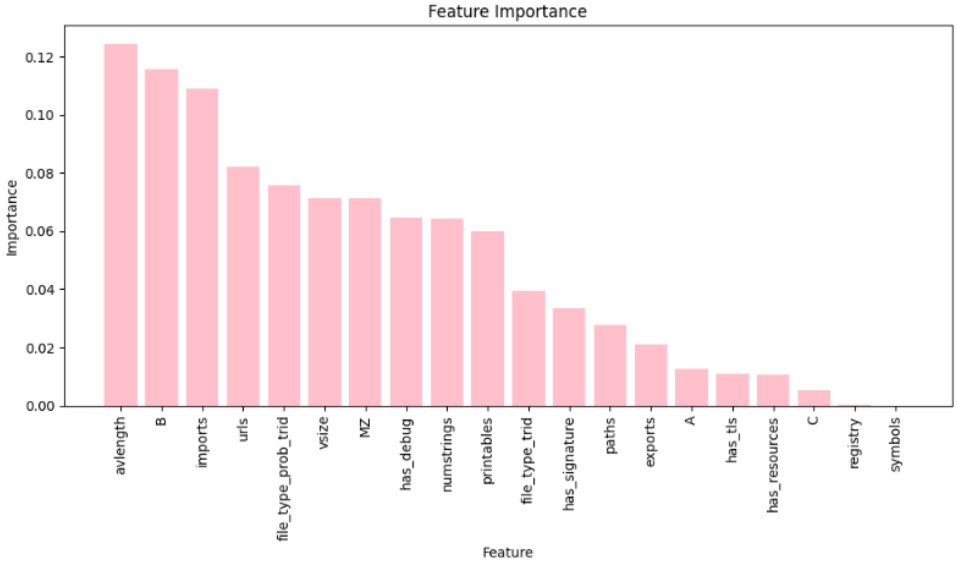








נספח 12 - גרף ברים המציג את חשיבות הפיצ'רים במודל Random Forest



נספח 13 – Confusion Matrix על סט ה -'train' וסט ה-'validation' של מודל Random Forest