**המרכז האקדמי רופין**

**בית-הספר להנדסה**

**המחלקה להנדסת חשמל ומחשבים**

**ספר פרויקט לשנת תש"פ**

**נושא הפרויקט:**

**Deep Learning for Malware Classification**

מבצעים:

עדן אביטן ת.ז 203286992

רועי אספורטס ת.ז 205718083

תומר עפרוני ת.ז 205711682

מנחה:

מנחה פנימי: ד"ר בני סלומון

השנה: תשפ״ב

תוכן עניינים

[1. תקציר 3](#_Toc116925324)

[2. מבוא 4](#_Toc116925325)

[2.1. הגדרת הבעיה 4](#_Toc116925326)

[2.2. שיטות עבודה וטכנולוגיות 4](#_Toc116925327)

[3. למידת מכונה ולמידה עמוקה 5](#_Toc116925328)

[3.1. למידת מכונה 5](#_Toc116925329)

[3.2. למידה עמוקה 6](#_Toc116925330)

[3.2.1. רשת עצבית מלאכותית 6](#_Toc116925331)

[4. מודלי למידה 7](#_Toc116925332)

[5. תהליך ביצוע המחקר 10](#_Toc116925333)

[6. תוצאות 10](#_Toc116925334)

[7. מסקנות 10](#_Toc116925335)

# תקציר

בעולמנו כיום עולה השימוש בטכנולוגיה בצורה כבירה ועם עלייה זו, צצים אין ספור איומים על המשתמשים השונים. מתקפות הסייבר בעולם מהוות סיכון מרכזי לפגיעה במרחב הקיברנטי של היעד במטרה לגנוב ממנו מידע ואף להסב לו נזק.

בנוסף לכך, תחום טכנולוגי אחר אשר מתפתח במהירות אדירה הינו הבינה המלאכותית – מצב שבו מנסים לדמות את יכולות החשיבה האנושיות ומבנה מוח האדם בעזרת אמצעים טכנולוגים. נושא זה מסתעף לתחומים רבים, אם כי הנושא העיקרי שמעניין אותנו הוא "הלמידה העמוקה" – תחום הנגזר מתוך למידת המכונה, אשר מתבסס על רשתות נוירונים (מודל מתמטי חישובי שפותח בהשראת תהליכים מוחיים/קוגניטיביים) בעלות מספר רב של שכבות.

פרויקט זה בא לשלב בין שני העולמות הללו. מטרתנו היא לדמות, לשחזר ואף להשוות את תוצאותינו לניסוי [1] שנערך באוניברסיטת סאן חוזה כחלק מתזה שבוצעה בתחום. הרעיון הכללי הוא בניית מודל למידה עמוקה אשר בהינתן תמונה של נוזקה (Malware) יידע לסווגה למשפחת נוזקות ספציפית.

תמונה של נוזקה – זוהי בעצם תוצאת המרה של נוזקה לכדי תמונה. כך נוכל לבצע סיווג על בסיס למידה עמוקה באמצעות עיבוד וניתוח התמונה.

בנוסף לכך, נרצה להשוות את תוצאות הסיווג כתוצאה משימוש ב-CNN לתוצאות סיווג שונות, המבוססות על מודלים שונים מעולמות למידת המכונה, אותם נבנה. זאת בכדי לנסות ולבדוק עלות אל מול תועלת (כגון: זמני ריצה, ביצועים וכו') בין מודלים פרימיטיביים אל מול מודלים חדישים כדוגמת CNN.

מכיוון שמדובר במודל סיווג נתונים, התוצאות אותן נרצה לספק יהיו הסתברויות שינועו בין 0 ל-1, כך שככל שנתקרב ל-1 אזי מדובר בזיהוי נכון.

בפרויקט זה בנינו ארבעה מודלים. שלושה מהם, הם מודלים פרימיטיביים שמטרתם להוות נקודת ייחוס למודל ה-CNN אותו רצינו לחקור, והם: LDA, QDA, GNB. כל אחד מאלה רץ (אימון ומבחן) על מאגר הנתונים כאשר כל מודל נחקר בשלושה אופנים שונים, כך שבכל אחד מהם התמונות במאגר הנתונים הוגדרו להיות:

32x32, 64x64, 128x128 פיקסלים.

התוצאות שהשגנו אכן מוכיחות את עליונות השימוש ב-CNN אל מול המודלים השונים שנבדקו בפרויקט זה.

בתוצאותינו ניתן לראות כי בתהליך המבחן על מאגר הנתונים 128x128 להוסיף השוואת נתונים ביחס לאחרים וביחס למחקר + מסקנות.

טודו:

# מבוא

תשתית החברה כיום מתבססת ברובה על מחשבים ואינטרנט. אבטחת הסייבר היא אומנות ההגנה על רשתות, מכשירים ונתונים מפני גישה בלתי מורשת או שימוש פלילי ובנוסף מהווה אבן יסוד בכל הנוגע להבטחת הסודיות, תקינות וזמינות המידע. לפי המחקר [1] עליו אנחנו מתבססים, בשנת 2018 הובחנו כ-669 מיליון סוגי נוזקות חדשות.

תחום העולה גם הוא בעולם הטכנולוגי בעקבות תוצאות וממצאים מחקריים טובים הוא תחום הבינה המלאכותית.

מתחום זה נגזרת הלמידה העמוקה אשר בה נעסוק בפרויקט זה. למידה עמוקה היא תחום העוסק באלגוריתמים ומודלים בהשראת מבנה ותפקוד המוח, מודלים אלו נקראים רשתות עצביות מלאכותיות.

ניתן לראות שילובים בין העולמות הללו בתוכנות אנטי וירוס שונות אשר מבססות את יכולת ההבחנה שלהן על בסיס בינה מלאכותית על מנת לאתר ולזהות פעולות ותהליכים זדוניים.

עקב העובדה כי תהליכים וקבצים זדוניים משתנים בקצב מהיר, קיים אתגר עבור עולם אבטחת הסייבר המתבסס על בינה מלאכותית לזהות ולנתר את אותם האיומים. בנוסף, מודלים ואלגוריתמים מבוססי למידה עמוקה מצריכים יכולות מחשוב גדולות על מנת לספק תוצאות בזמן סביר.

* 1. הגדרת הבעיה

הבעיה אותה הפרויקט בא לפתור היא בעיית סיווג נוזקות למשפחות באמצעות למידת מכונה. כלומר, סיווג שיטתי של איומים אלו, המאיימים על מערכות תקשורת ואינטרנט המהוות תשתית עבור ארגונים רבים. ביצוע והצלחה בפתירת בעיה זו, כרוך בקשיים רבים והתאמות משמעותיות.

* 1. שיטות עבודה וטכנולוגיות
     1. סביבת העבודה שבה השתמשנו היא Jupyter Notebook אשר תרוץ על Google Colab.
     2. הפרויקט כולו נכתב ב-Python, כאשר השתמשנו במגוון ספריות שונות, כגון:
* Keras אשר שימשה ליצירת מודל ה-CNN.
* Scikit-Learn אשר שימשה ליצירת המודלים LDA, QDA, GNB.
* Pandas & Numpy אשר סייעו בעיבוד המידע.
  + 1. מאגר הנתונים שעליו התבססנו הוא Malimg Dataset המכיל כ-9764 תמונות בסקלה של גווני הצבע האפור. את המאגר נמיר לשלושה גדלים שונים (פיקסלים) והם: 32x32, 64x64, 128x128. זאת על ידי שימוש בספריית Pillow.
    2. הפרויקט וקוד המקור מנוהלים באמצעות GitHub.

1. למידת מכונה ולמידה עמוקה
   1. למידת מכונה

תת-תחום במדעי המחשב ובבינה מלאכותית. התחום עוסק בפיתוח אלגוריתמים המיועדים לאפשר למחשב ללמוד מתוך דוגמאות, ופועל במגוון משימות חישוביות בהן התכנות הקלאסי אינו אפשרי.

תחום זה מכיל שלושה סוגי למידה עיקריים והם:

* למידה מונחית (Supervised Learning): מאופיינת על ידי תיוג מאגר המידע לקטגוריות, כך שבתהליך הלמידה של המערכת היא תוכל להשוואת את תוצאותיה לתיוג שהוגדר מראש, ובעצם ללמוד לסווג את המידע על סמך תשובות שהוגדרו, תהליך זה יקרא שלב האימון (Train). לבסוף, המערכת תידרש לסווג קלטים חדשים שאיננה מכירה לקבוצה המתאימה, תהליך זה יקרא שלב הבדיקה (Test).
* למידה בלתי מונחית (וUnsupervised Learning): מאופיינת על ידי היכולת לקבל מידע שאינו מתויג לקטגוריות, ובעצם לדעת לנתח ולאגד אותו ללא התערבות אדם.
* למידת חיזוק (Reinforcement Learning): מאופיינת על ידי היכולת להעניק תגמול על התנהגות נכונה ו-״להעניש״ כאשר ההתנהגות אינה נכונה. המערכת תקבל משוב חלקי בסיום העבודה ותדע להסיק מסקנות לגבי השלבים אשר אותם יש לשפר/לשמר.

פרויקט זה מתמקד בעיקרו בלמידת מונחית, כאשר תייגנו את התמונות ל-25 משפחות שונות של נוזקות.

בעת אימון מודלי למידת מכונה, יש להיזהר ממצב של התאמת יתר (Overfitting), כמתואר באיור (1), אשר עלול לגרום לכך שהמערכת תדע לסווג בצורה טובה את הדאטה שניתן לאימון, אך לא תצליח לעשות זאת עבור מידע חדש.



איור 1: הצגת קו חלוקה בצורות התאמה שונות

* 1. למידה עמוקה

היא חלק ממשפחה רחבה יותר של שיטות למידת מכונה המבוססות על רשתות עצביות מלאכותיות בשילוב עם למידת מאפיינים. הרעיון מאחורי שיטה זו היא היכולת ללמד מחשבים לבצע עבודה אשר נראית טבעית למוח האדם אך מסובכת לביצוע המחשב.

בלמידה עמוקה המודל יקבע אילו מהפיצ׳רים רלוונטיים להסקת המסקנות ואילו לא, לעומת למידת מכונה, אשר בה יש לבצע עבודת הכנה מוקדמת של סידור הדאטה ובחירת הפיצ׳רים המתאימים.

במקרה שלנו, הפיצ׳רים באמצעותם נרצה לפתור את בעיית הסיווג הם הפיקסלים של התמונות אותם נכניס כקלט למודלים.

* + 1. רשת עצבית מלאכותית

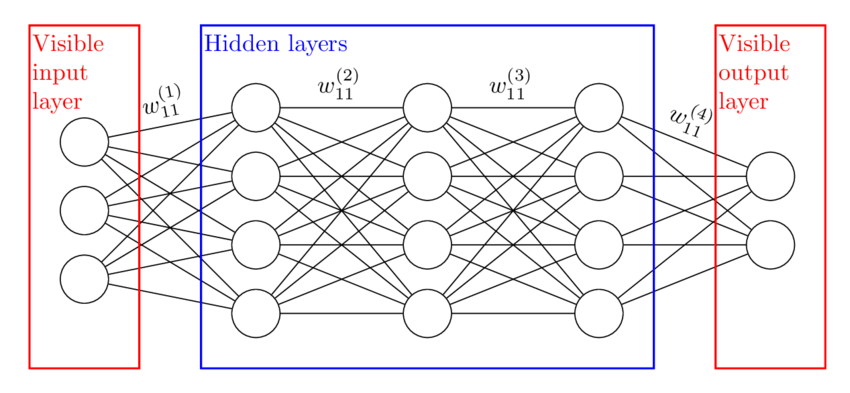
רשת עצבית מלאכותית (ANN – Artificial Neural Network) היא מודל מתמטי חישובי, שפותח בהשראת תהליכים מוחיים או קוגניטיביים המתרחשים ברשת עצבית טבעית ומשמש במסגרת למידת מכונה. הרשת מכילה לרוב מספר גדול של יחידות מידע המקושרות זו לזו, כאשר צורת הקישור בין היחידות מזכירה את אופן חיבור הנוירונים במוח ומכאן נובע שמה.

ברשת מספר רב של נוירונים כאשר כל אחד מהם מבצע חישוב פשוט יחסית. אל הנוירון מתקבל קלט אשר עליו מבוצע עיבוד באמצעות נוסחה מוגדרת שתוצאתה מהווה הפלט של אותו הנוירון. פלט זה מועבר הלאה אל נוירונים אחרים ברשת אשר מבצעים תהליך דומה.

הרשת מאופיינת על ידי מספר תכונות כגון, צורת חיבור הנוירונים המהווה האופן בו הנוירונים מחוברים האחד לשני בין השכבות השונות, פונקציית ההפעלה (Activation Function) הקובעת האם קלט הנוירון ברשת הוא חשוב או לא וזאת באמצעות פעולות מתמטיות, ומשקלי נוירונים אשר קובעים עבור כל נוירון את חשיבותו ביחס למודל עצמו ולהמשך התהליך.

רשת הנוירונים מורכבת משלוש שכבות עיקריות (איור 2) – שכבת הקלט, שכבת העיבוד ושכבת הפלט.

* שכבת הקלט (Input Layer): שכבה זו מקבלת את המידע הראשוני אשר אמור להיקלט אל תוך המודל. אל כל נוירון השייך לשכבה זו מתקבל ערך מידע יחיד. משכבה זו עובר המידע אל השכבה הבאה בתור – השכבה החבויה (Hidden Layer).  
  במקרה שלנו, שכבת הקלט היא הפיקסלים של התמונה, לפיכך עבור מודל של מספר הנוירונים בשכבת הכניסה תהיה . בנוסף, כל פיקסל יהווה מאפיין (Feature) עבור המודל.
* שכבה חבויה (Hidden Layer): שכבה זו ממוקמת בין שכבת הקלט ושכבת הפלט. מטרת השכבה היא ביצוע חישובים וטרנספורמציות לא לינאריות על הנתונים אשר נקלטו בה. לבסוף, שכבה זו תעביר את תוצאותיה אל שכבת הפלט (Output layer).  
  במקרה שלנו, השכבה החבויה תורכב משכבות קונבולוציה שונות (אשר יפורטו בהמשך) כדי לבצע חישובים מתאימים.
* שכבת הפלט (Output Layer): השכבה הסופית ברשת אליה מתקבלים כלל התחזיות והתוצאות מהמודל.  
  במקרה שלנו, תוצאות המודל אשר יתקבלו בשכבה זו יהיו מספרים בין אפס לאחד, אשר יהוו הסתברות של הקלט להיות שייך למשפחת נוזקות מסוימת. שכבת הפלט שלנו מורכבת מ-25 נוירונים אשר כל אחד מהם מהווה משפחת נוזקות שונה.



איור 2: מבנה רשת נוירונים

1. מודלי למידה

מודל למידת מכונה הינו אלגוריתם אשר מטרתו היא פתרון בעיה כלשהי. במקרה שלנו, בעיה זו היא בעיית הסיווג. קיימים מספר אלגוריתמים ושיטות פתרון. שיטות אלה בכלליותם מנסות, בעזרת כלים מתמטיים וסטטיסטיים, למצוא דפוסי התנהגות בתופעות כלשהן אשר יסייעו בפתרון הבעיה. את תופעות ובעיות אלה ניתן לתאר במגוון צורות, אך כאשר אנו רוצים להמיר אותן לכדי מודל מתמטי אותו ניתן לנסות לפתור בשיטות הידועות, יש לתאר אותן באותם מונחים. מונחים אלו יכולים להיות:

* מאפייני התופעה (Features) אשר בחלקם יכולים להיות פרמטרים כמותיים ובחלקם האחר, פרמטרים שהינם יותר אבסטרקטיים אשר צריך לקבוע כיצד ובאיזה אופן יש להמירם לכדי משתנים כמותיים. בחלק מהמקרים, אנו קבועים אלו מהמאפיינים רלוונטיים עבור המודל ולפיכך גם מספקים לו את מאפיינים אלו. במקרים אחרים, נספק למודל את כלל המאפיינים וניתן לו לקבוע בעצמו אילו מהמאפיינים רלוונטיים עבורו.
* פלט המודל, אשר מהווה את פתרון הבעיה. במקרים מסוימים פתרון הבעיה יכול להיות משתנה בדיד אשר יכול לייצג הסתברות כלשהי, ובמקרים אחרים מספר משתנים אשר כל אחד מהם מאפיין תוצאה אפשרית אחרת שיכולה להתקבל ביחס לקלט הנתון. קיימים מגוון רחב של אפשרויות נוספות, כאשר כל אפשרות נקבעת בהתאם לסוג הבעיה.
  1. QDA (Quadratic Discriminant Analysis)

זהו מודל/אלגוריתם למידת מכונה העוסק בבעיית הסיווג. כלומר, מטרתו לסווג בין 2 מחלקות (Classes) או יותר, באופן כזה שעבור כל מחלקה, לבסוף, מותאם גאוסיאן אשר תוחם בתוכו את כלל הערכים האפשריים עבור מחלקה זו כמתואר באיור (3).

מודל זה עובד על ידי חישוב מאפיינים סטטיסטים על הנתונים המסופקים לו. עבור משתנה קלט בודד יוגדרו הממוצע והשונות שלו עבור כל מחלקה (Class). כאשר מדובר משתני כניסה מרובים, קיים ייצוג מטריציוני באופן דומה.

תוצאות חישוב סטטיסטי זה משמשים לבסוף בכדי לייצג את המודל המדובר ולבסוף גם ישמשו לביצוע תחזיות בהמשך.

הנחות יסוד אשר מודל זה מניח על המידע המתקבל אליו:

* הנתונים מתפלגים נורמלית.
* לכל מחלקה יש שונות משותפת (Covariance) אחרת אשר מאפיינת אותה.

נגדיר את חישוב הערך הממוצע () לכל קלט () עבור כל מחלקה () באופן הבא:

נגדיר שונות למחלקה באופן הבא:

כאשר זהו מספר המופעים של אותה מחלקה.

* 1. LDA (Linear Discriminant Analysis)

מודל זה מתבסס ברובו על מודל ה-QDA, אם כי השוני העיקרי ביניהם הוא שבניגוד ל-QDA, LDA משתמש בשונות משותפת זהה עבור כלל המחלקות הקיימות.

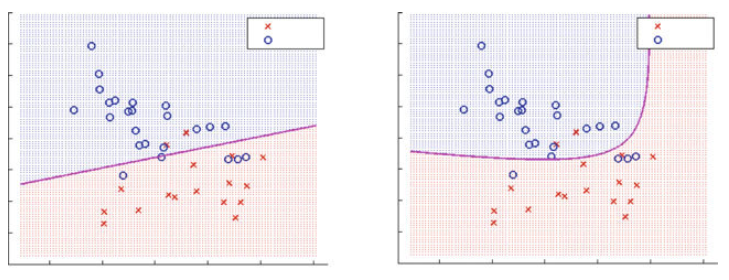
הנחות יסוד אשר מודל זה מניח על המידע (המידע המתקבל למודל):

* הנתונים מתפלגים נורמלית.
* לכל מחלקה יש את אותה השונות המשותפת.

יתר החישובים מתבצעים באופן דומה לחישובי QDA מלבד ההנחות המצוינות לעיל.

LDA

QDA



איור 3: LDA Vs QDA

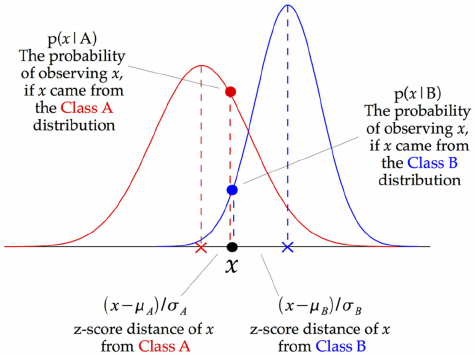
* 1. GNB (Gaussian Naive Bayes)

גם מודל זה מתבסס ברובו על מודל ה-QDA. השוני העיקרי בין שני המודלים הוא שמודל ה-GNB מניח כי מטריצות השונות המשותפת המתקבלות הן מטריצות אלכסוניות.

הנחות יסוד אשר מודל זה מניח על המידע (המידע המתקבל למודל):

* הנתונים מתפלגים נורמלית.
* לכל מחלקה יש מטריצת שונות משותפת אחרת אשר מאפיינת אותה, כאשר מטריצה זו היא מטריצה אלכסונית.
* אין תלות סטטיסטית בין ערכי האלכסון (בין הקלאסים השונים).

יתר החישובים מתבצעים באופן דומה לחישובי QDA מלבד ההנחות המצוינות לעיל.

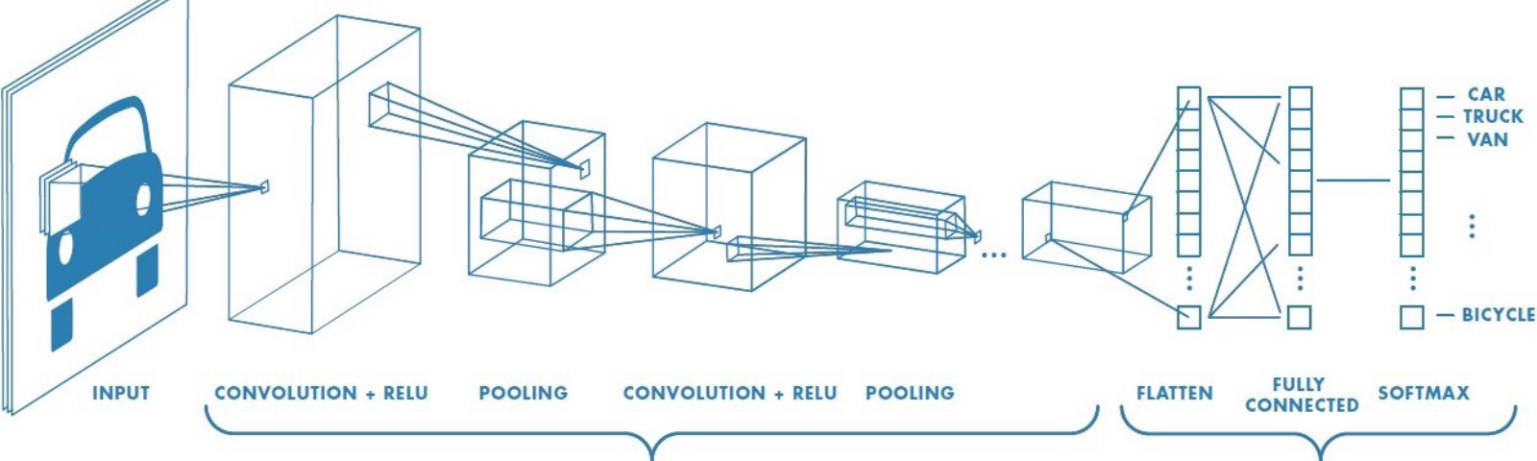


איור 4 : GNB

* 1. CNN (Convolutional Neural Network)

זוהי סוג של רשת נוירונים המשמשת בעיקר לניתוח תמונות. הרשת מבוססת על שכבות כינוס (Convolution) הממומשות על ידיי העברת פילטר על גביי הקלט ומשתמשת בפעולת הקונבולוציה (לפחות בשכבה אחת). ברשת CNN, בדרך כלל נתייחס לצבעי התמונה המשמשת כקלט כמערך תלת מימדי כאשר כל מערך בו מאופיין על ידי אחד משלושת הצבעים: אדום, ירוק וכחול (RGB). בנוסף, מקדמי הקונבולוציה נלמדים עצמאית במהלך אימון המודל.

כאשר משתמשים ב-CNN על תמונות, נקבל עמידות רבה (Invariance) ביחס לטרנספורמציות על נתונים אשר תעזור להקטין את הסיכויים לקבלת overfitting אשר עלולים להגרם עם ריצת המודל.



* + 1. קונבולוציה

במקרה שלנו נשתמש בקונבולוציה בדידה אשר מהווה פעולה מתמטית של סכימת מכפלות האיברים בין שתי מטריצות כמתואר באיור (5).



איור 5: קונבולוציה

* + 1. פילטר (Filter)

חלון מטריציוני בגודל מסוים המכיל תבנית שבעזרתה מתבצעת פעולת הקונבולוציה על הקלט, כאשר המטרה הסופית היא למצוא דמיון בין התבנית לבין אזורים שונים על גבי התמונה. למשל, קווים ישרים, עיקולים, צורות, דפוסים וכו'. ככל שהרשת יותר עמוקה, כך המודל יכול לזהות אובייקטים יותר מורכבים.

טודו: להוסיף גם סוגי שכבות, אולי פיצ'ר מאפ (קודם זה) + תמונה לפילטר.

1. תהליך ביצוע המחקר

טודו: להסביר על KFOLD ומה זה K

1. תוצאות

בכל אחד מהמקרים, נציג את תוצאות המודלים כאשר נבצע חלוקה למקרים ביחס לשכבת הקלט המאופיינת על ידי כמות הפיקסלים של כל תמונה במאגר התמונות הנבדק.

טודו: עבור כל K להוסיף את חלוקת התמונות (הדאטה סט) לטריין טסט

* 1. QDA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Size | Best K - Test | Test Accuracy | Best K - Train | Train Accuracy |
| 32x32 | 10 | 58.224 | 2 | 96.815 |
| 64x64 | 4 | 54.915 | ALL | 100 |
| 128x128 | 7 | 48.976 | ALL | 100 |

טבלה 1: תוצאות QDA

Chart, line chart

Description automatically generated

**Train (Includes test results)**

K = 2

Success Rate (after normalizing):

- Train: 96.815 %

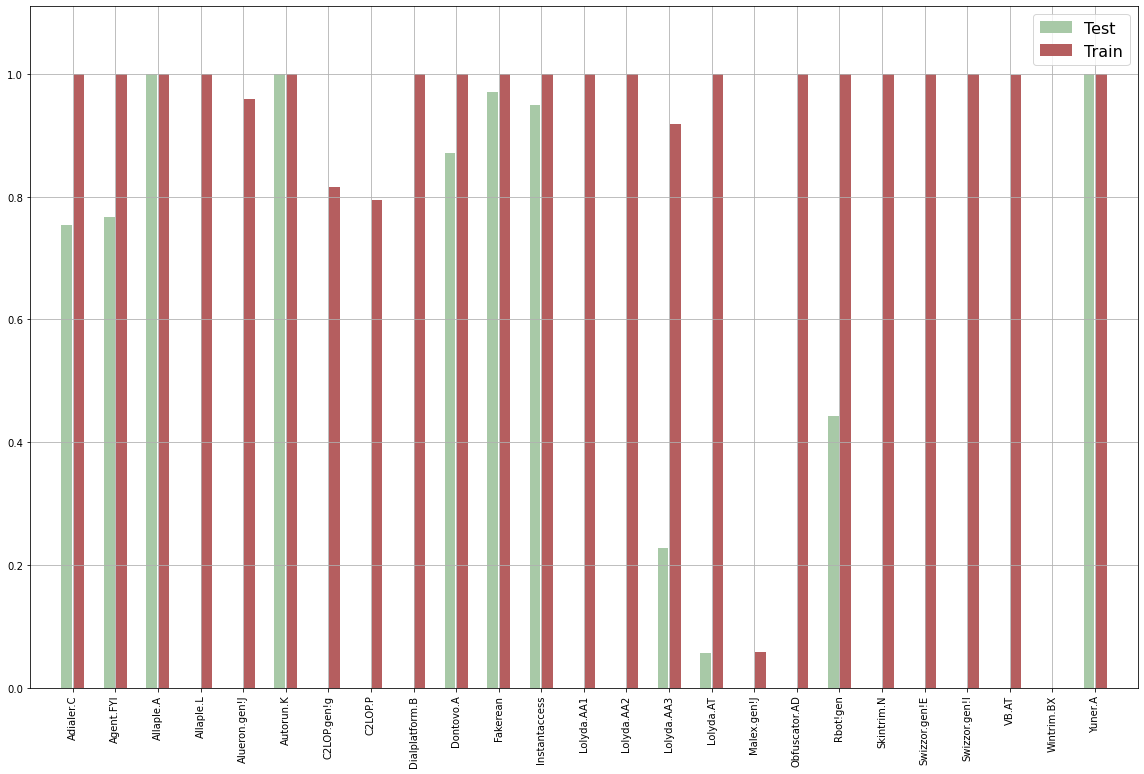
- Test: 55.653 %

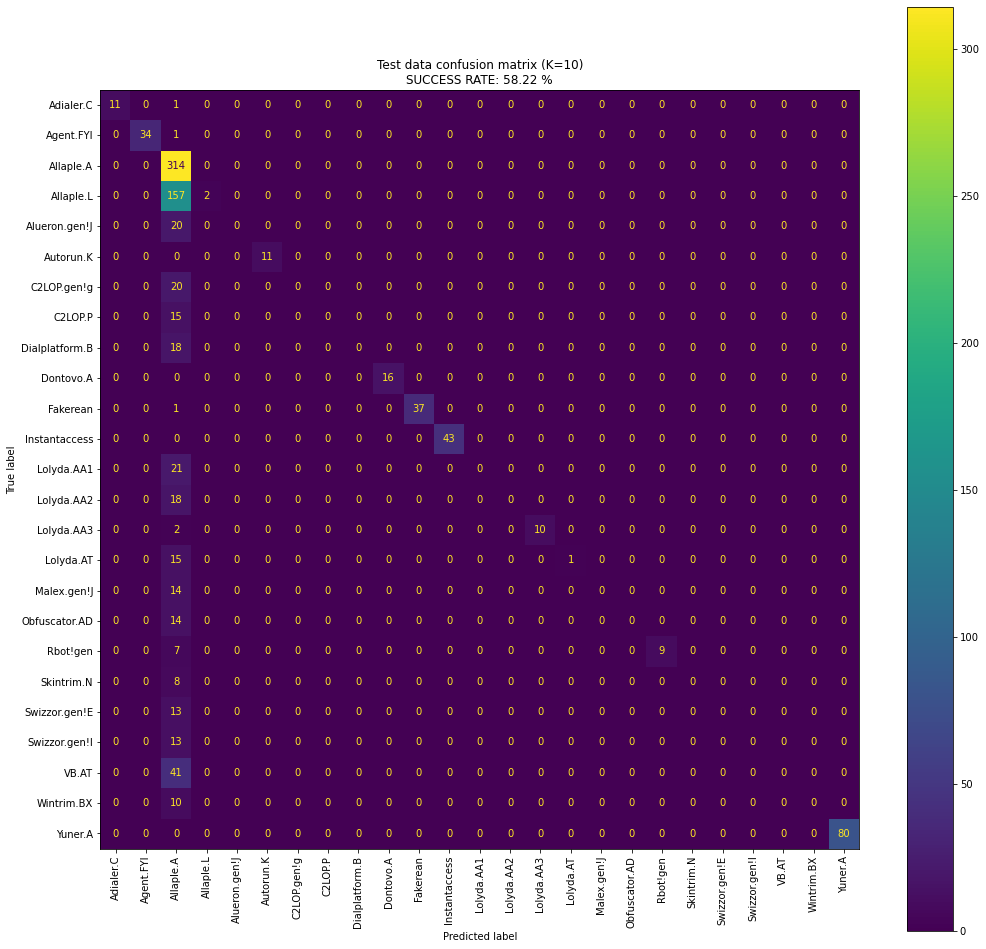
Malware family with the highest false classification (after normalizing):

- Train: Wintrim.BX

- Test: Allaple.LA picture containing text, monitor, electronics, display

Description automatically generated

****



**Test (Includes train results)**

K = 10

Success Rate (after normalizing):

- Train: 93.821 %

- Test: 58.224 %

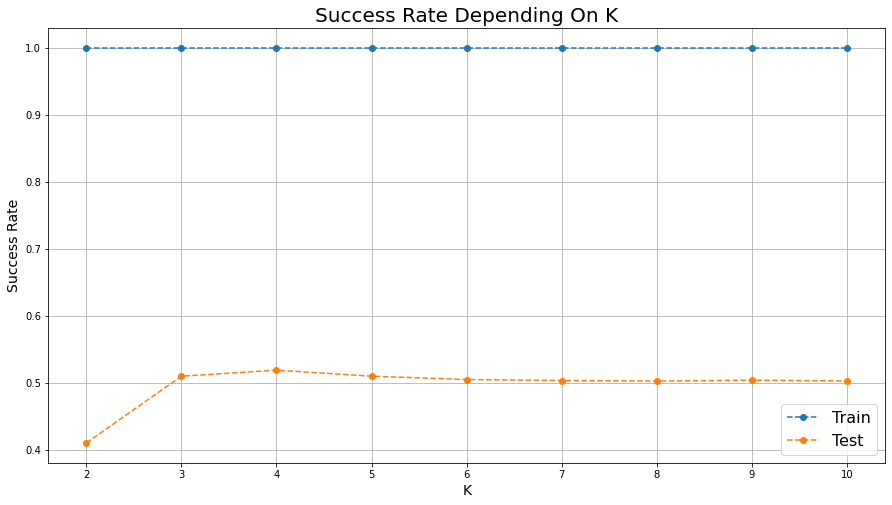
Malware family with the highest false classification (after normalizing):

- Train: Wintrim.BX

- Test: Alueron.gen!J

Chart, bar chart

Description automatically generated



**Train (Includes test results)**

K = ALL

Success Rate (after normalizing):

- Train: 100.000 %

Malware family with the highest false classification (after normalizing): N/A

**Test (Includes train results)**

K = 4

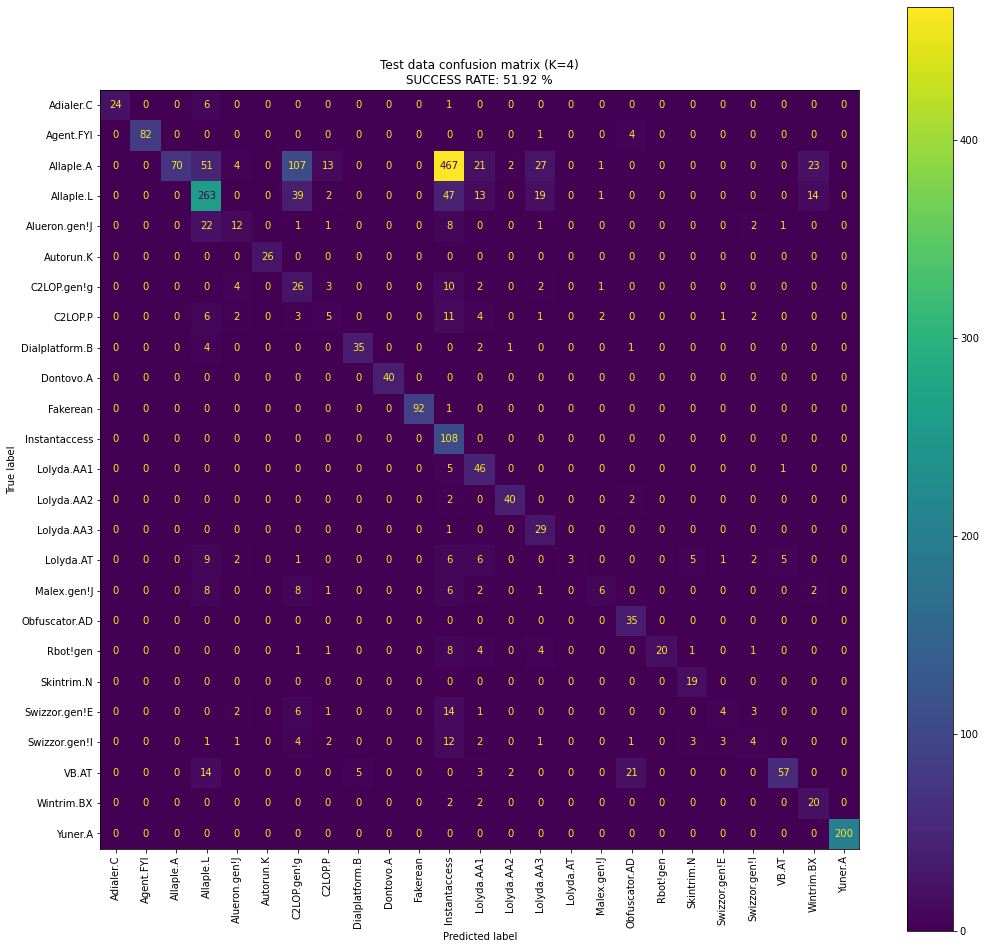
Success Rate (after normalizing):

- Train: 100.000 %

- Test: 51.915 %

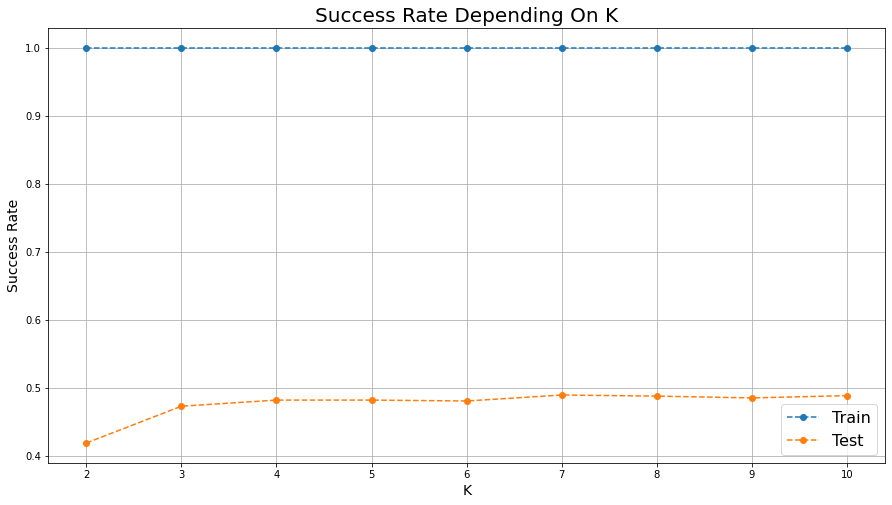
Malware family with the highest false classification (after normalizing):

- Train: N/A

 - Test: Lolyda.AT

Background pattern

Description automatically generated



**Train (Includes test results)**

K = ALL

Success Rate (after normalizing):

- Train: 100.000 %

Malware family with the highest false classification (after normalizing): N/A

**Test (Includes train results)**

K = 7

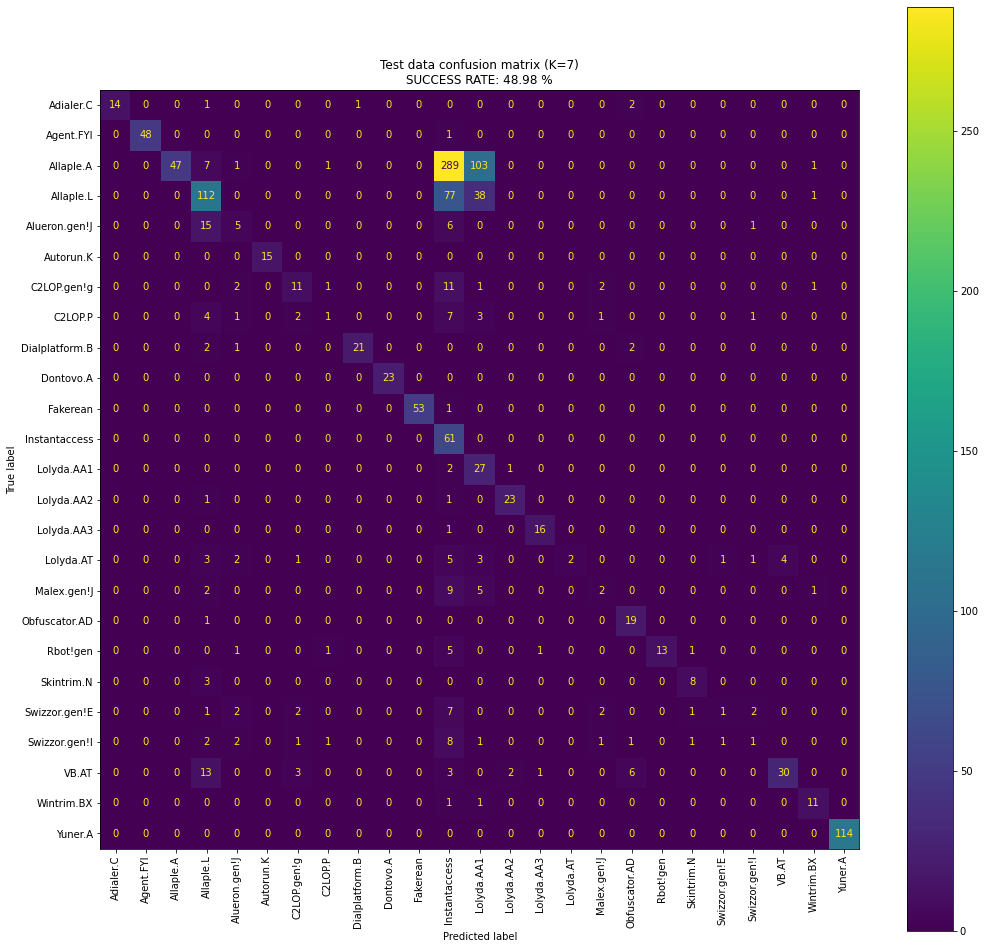
Success Rate (after normalizing):

- Train: 100.000 %

- Test: 48.976 %

Malware family with the highest false classification (after normalizing):

- Train: N/A

 - Test: Swizzor.gen!I

* 1. LDA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Size | Best K - Test | Test Accuracy | Best K - Train | Train Accuracy |
| 32x32 | 6 | 97.009 | 2 | 99.15 |
| 64x64 | 10 | 97.347 | 2 | 100 |
| 128x128 | 3 | 70.73 | ALL | 100 |

טבלה 2: תוצאות LDA

Chart, line chart

Description automatically generated

**Train (Includes test results)**

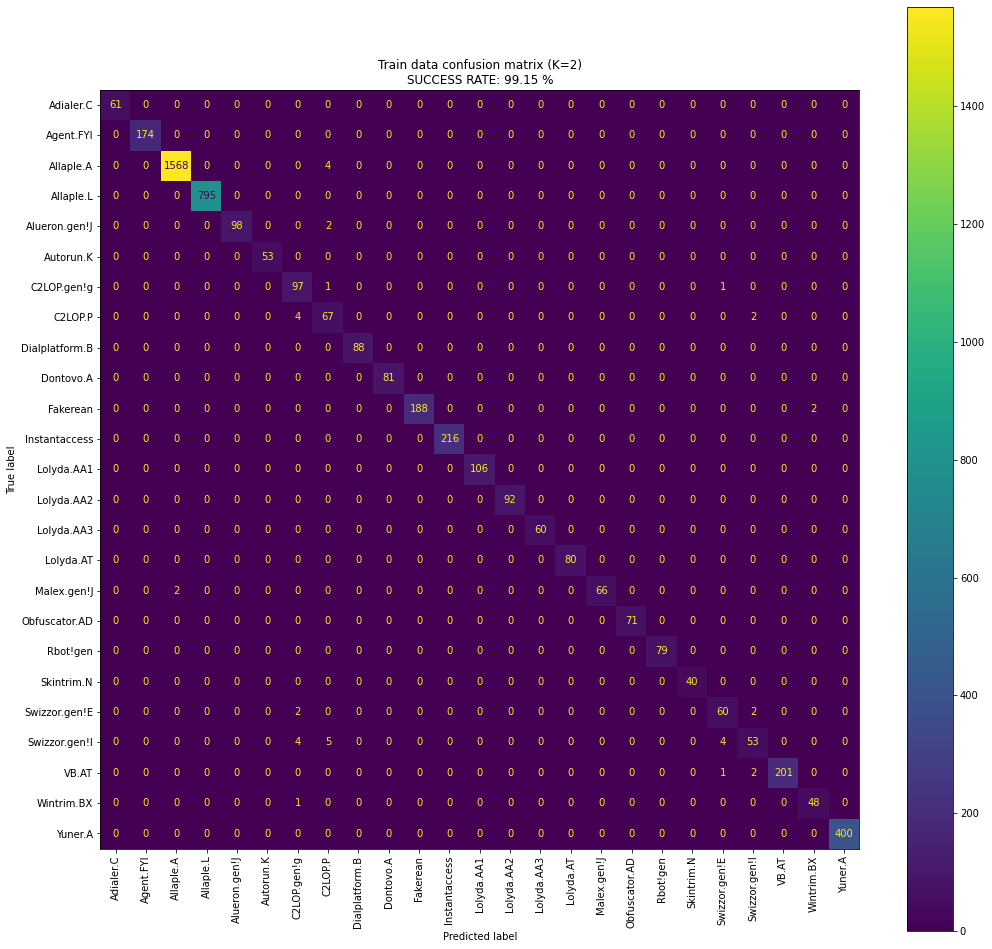
K = 2

Success Rate (after normalizing):

- Train: 99.150 %

- Test: 96.846 %

Malware family with the highest false classification (after normalizing):

 - Train: Swizzor.gen!I

- Test: Swizzor.gen!I

Background pattern

Description automatically generated

**Test (Includes train results)**

K = 6

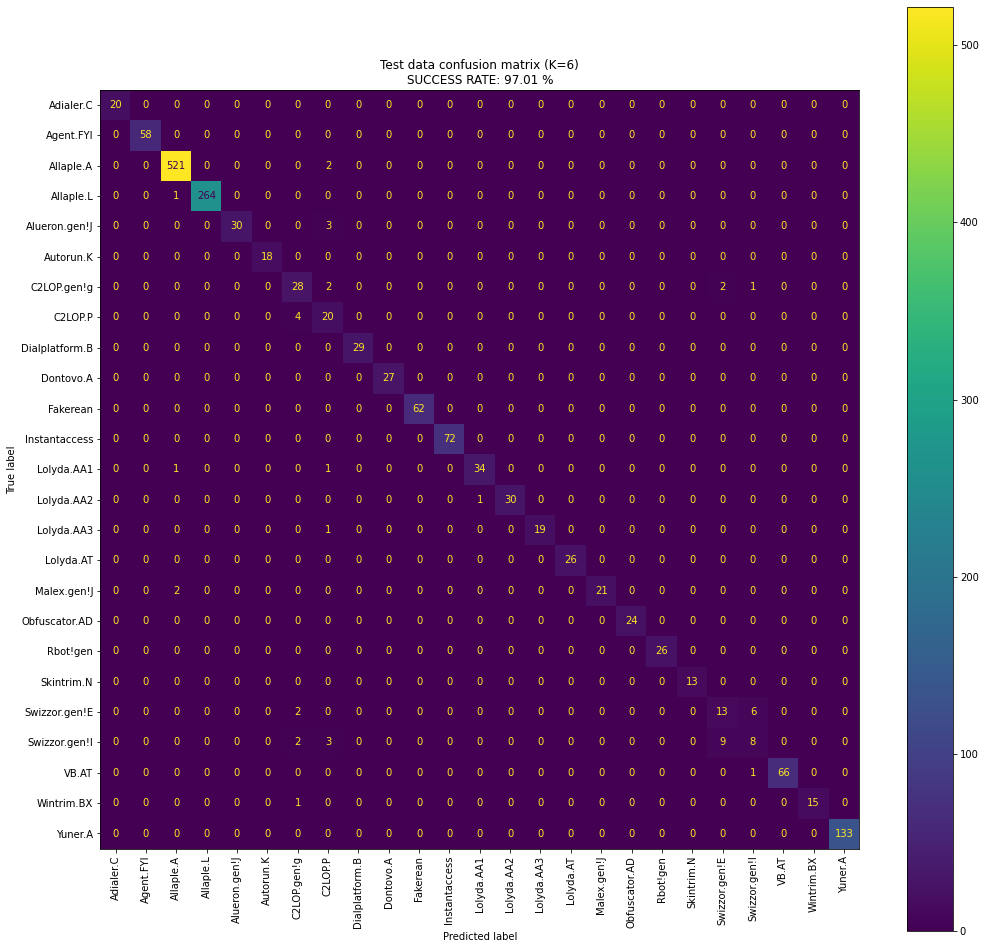
Success Rate (after normalizing):

- Train: 98.417 %

- Test: 97.009 %

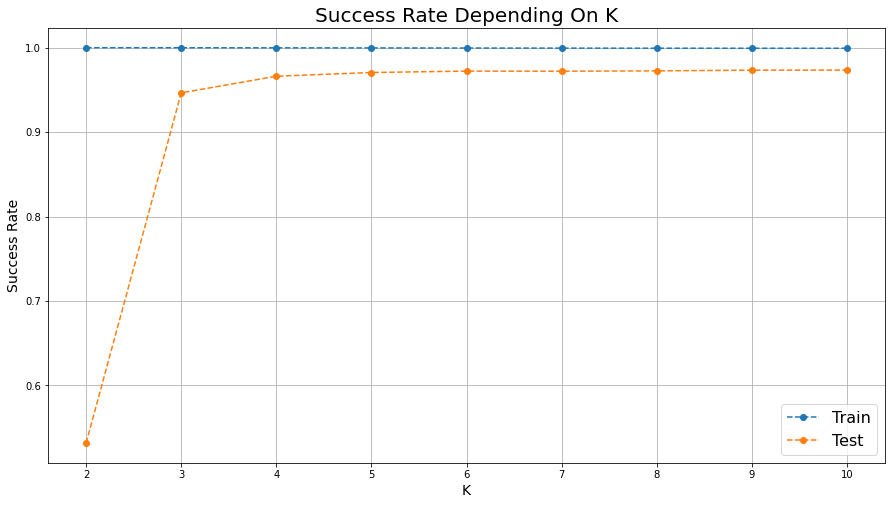
Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I

 - Test: Swizzor.gen!I

Chart, background pattern, bar chart

Description automatically generated



**Train (Includes test results)**

K = 2

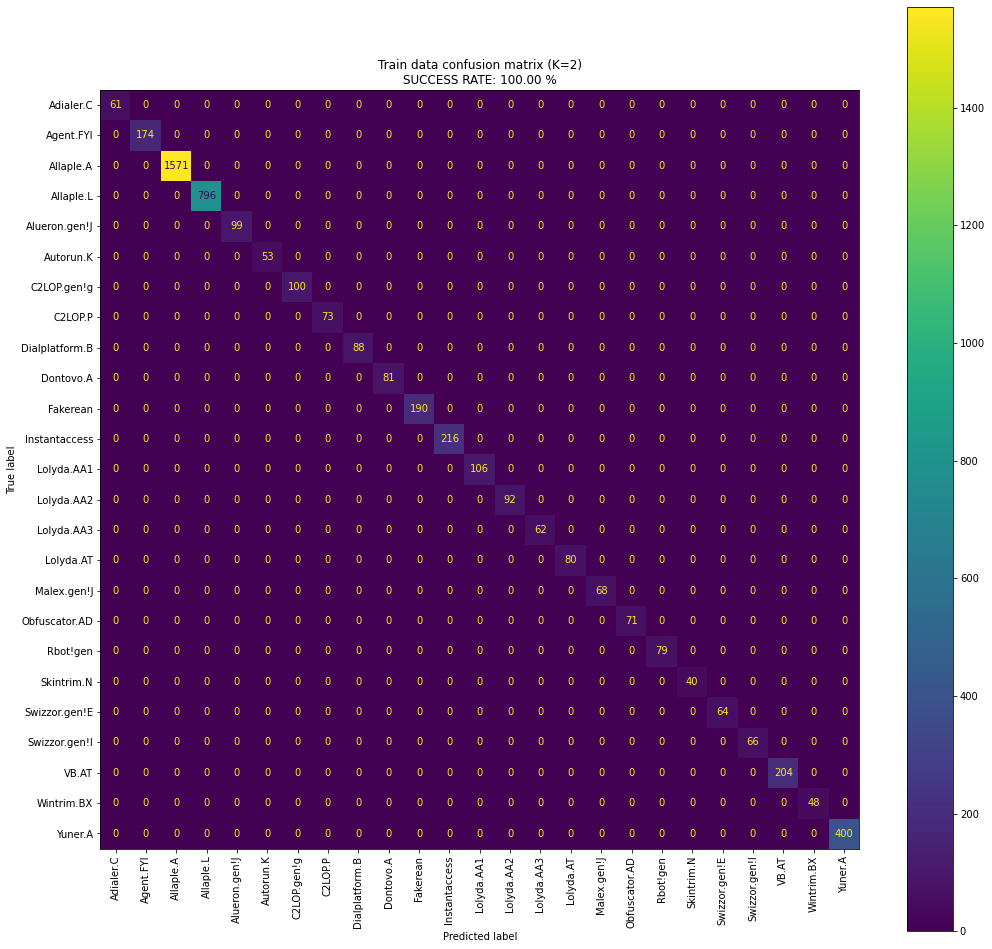
Success Rate (after normalizing):

- Train: 100.000 %

- Test: 53.175 %

Malware family with the highest false classification (after normalizing):

- Train: N/A

 - Test: Swizzor.gen!E

Background pattern

Description automatically generated

**Test (Includes train results)**

K = 10

Success Rate (after normalizing):

- Train: 99.933 %

- Test: 97.347 %

Malware family with the highest false classification (after normalizing):

- Train: Lolyda.AA3

A picture containing text, monitor, electronics, display

Description automatically generated - Test: Swizzor.gen!I

Chart, line chart

Description automatically generated

**Train (Includes test results)**

K = ALL

Success Rate (after normalizing):

- Train: 100.000 %

Malware family with the highest false classification (after normalizing): N/A

**Test (Includes train results)**

K = 3

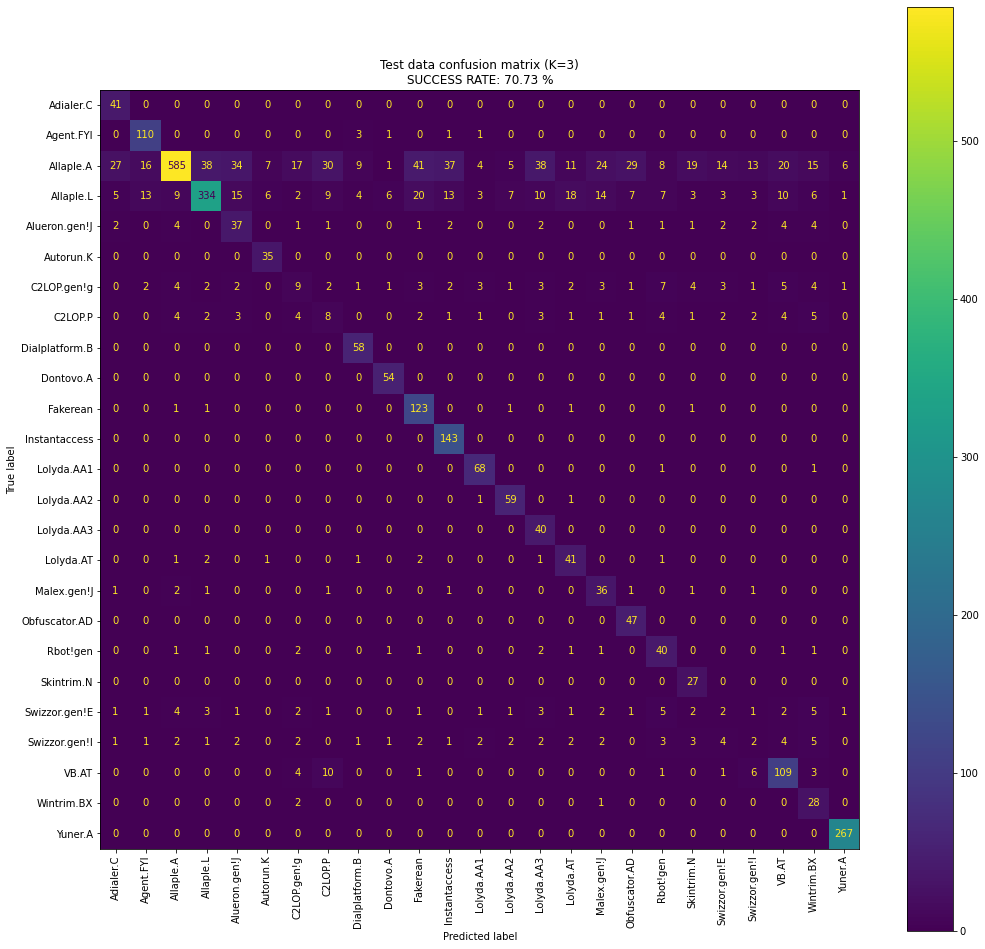
Success Rate (after normalizing):

- Train: 100.000 %

- Test: 70.730 %

Malware family with the highest false classification (after normalizing):

- Train: N/A

 - Test: Swizzor.gen!I

**Background pattern

Description automatically generated**

* 1. GNB

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Size | Best K - Test | Test Accuracy | Best K - Train | Train Accuracy |
| 32x32 | 9 | 95.258 | 2 | 95.678 |
| 64x64 | 3 | 96.467 | 2 | 97.111 |
| 128x128 | 3 | 96.887 | 2 | 97.634 |

טבלה 3: תוצאות GNB

**Chart, line chart

Description automatically generated**

**Train (Includes test results)**

K = 2

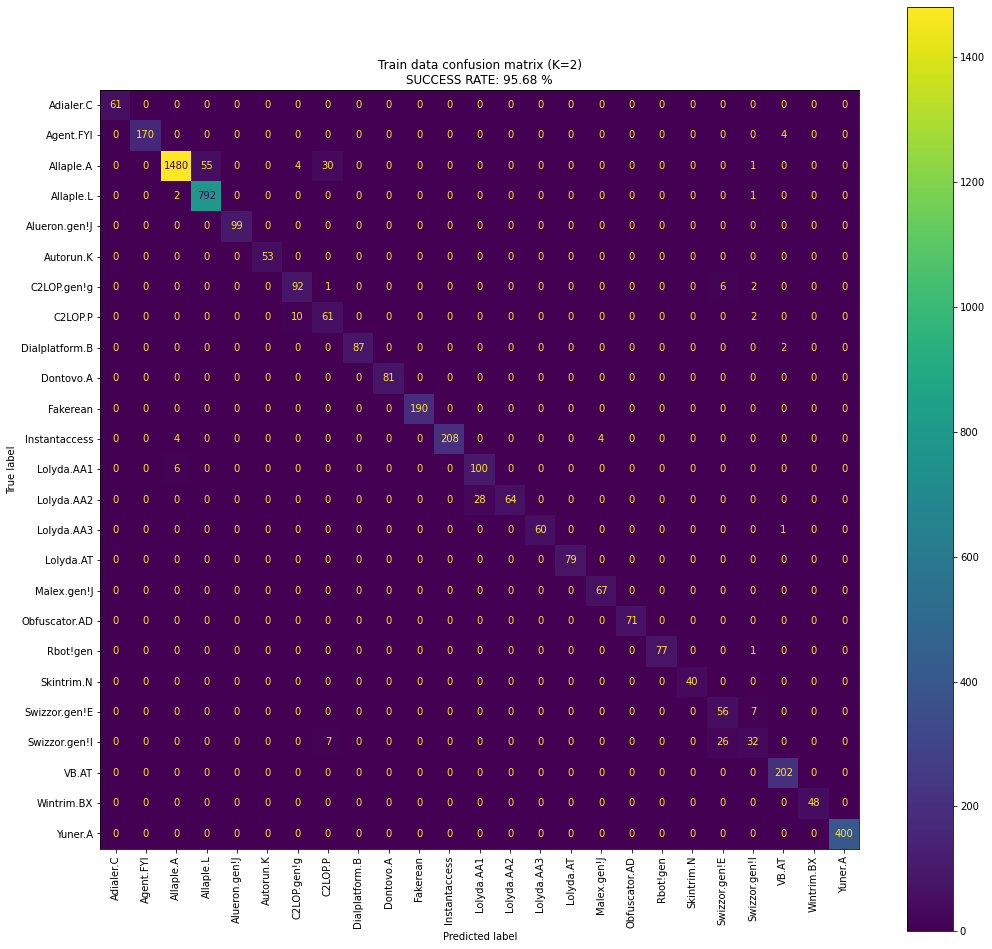
Success Rate (after normalizing):

- Train: 95.678 %

- Test: 94.982 %

Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I

 - Test: Swizzor.gen!I

**Chart, bar chart

Description automatically generated**

**Test (Includes train results)**

K = 9

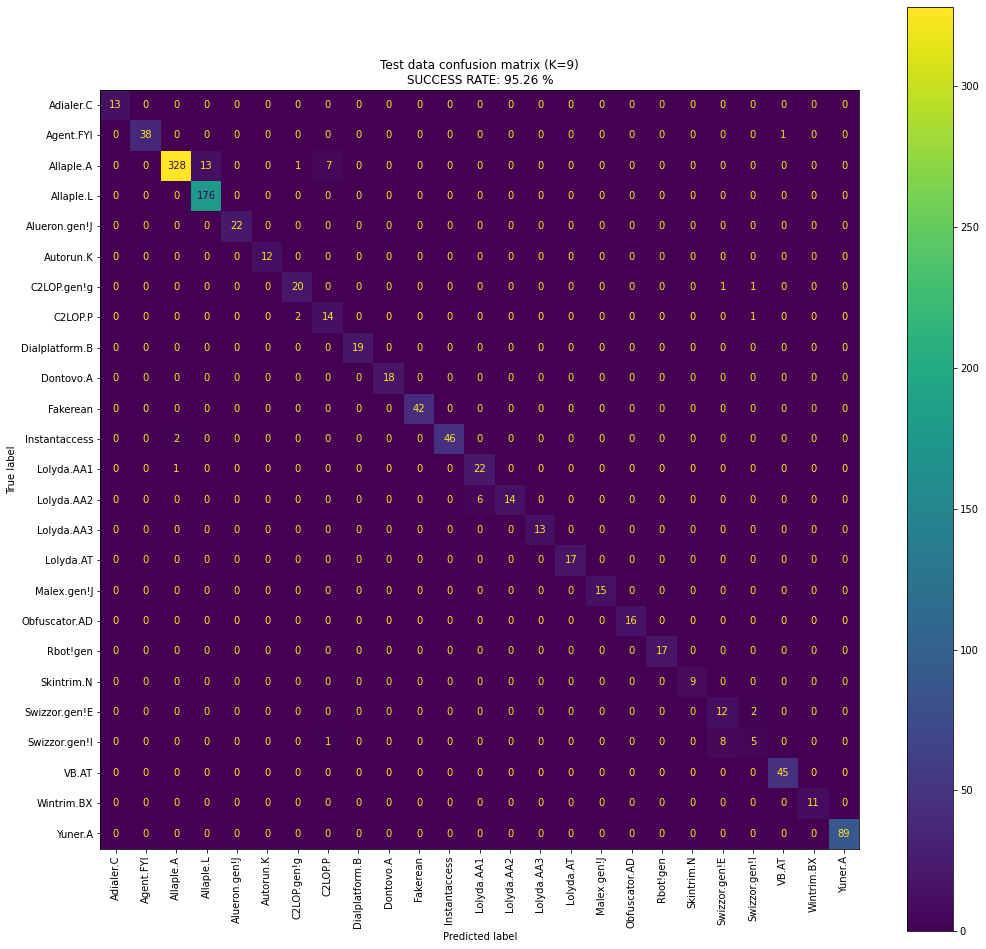
Success Rate (after normalizing):

- Train: 95.428 %

- Test: 95.258 %

Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I

 - Test: Swizzor.gen!I

Chart, bar chart

Description automatically generated

**Chart, line chart

Description automatically generated**

**Train (Includes test results)**

K = 2

Success Rate (after normalizing):

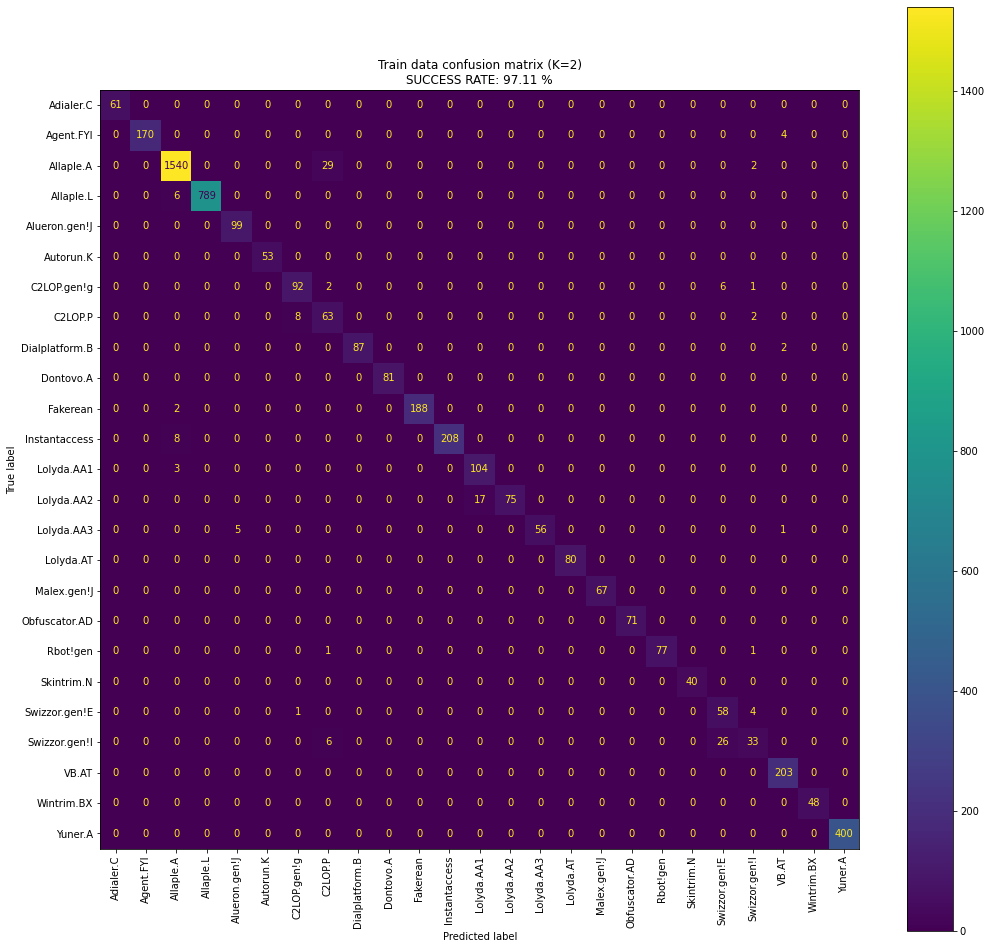
- Train: 97.112 %

- Test: 96.221 %

Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I

- Test: Swizzor.gen!I

****

**Chart, background pattern, bar chart

Description automatically generated**

**Test (Includes train results)**

K = 3

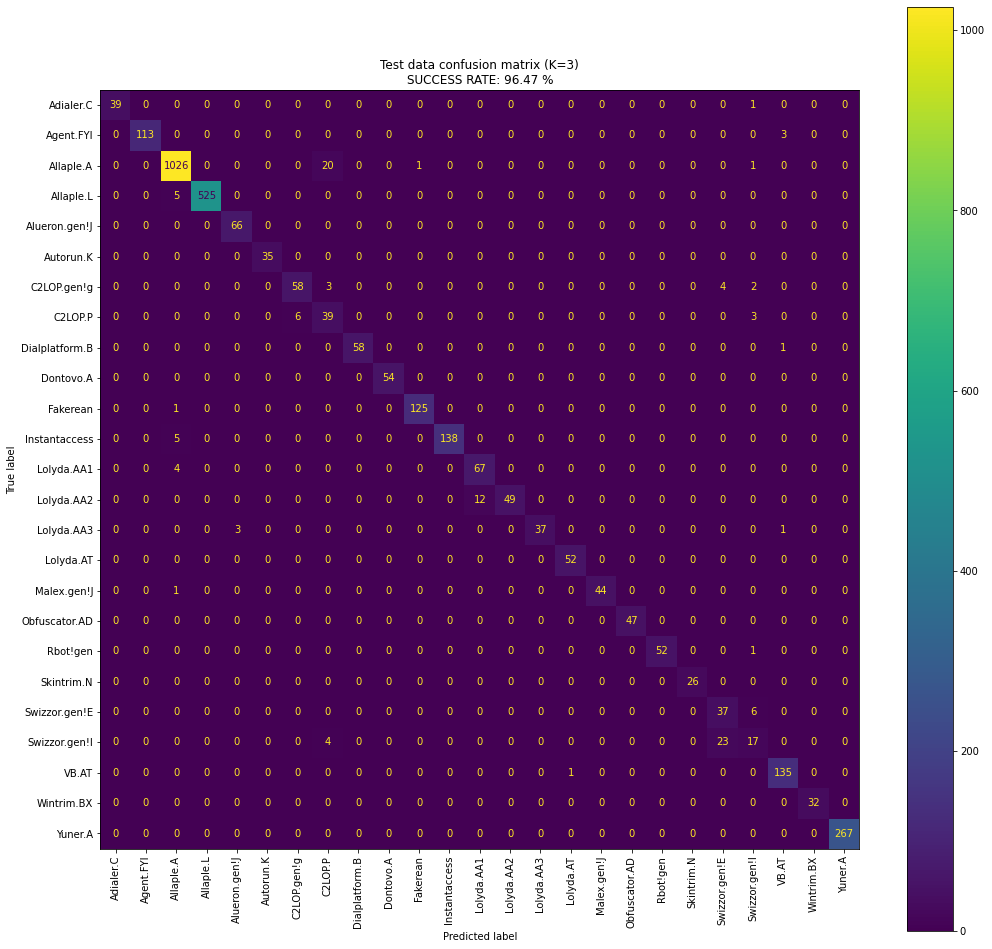
Success Rate (after normalizing):

- Train: 96.881 %

- Test: 96.467 %

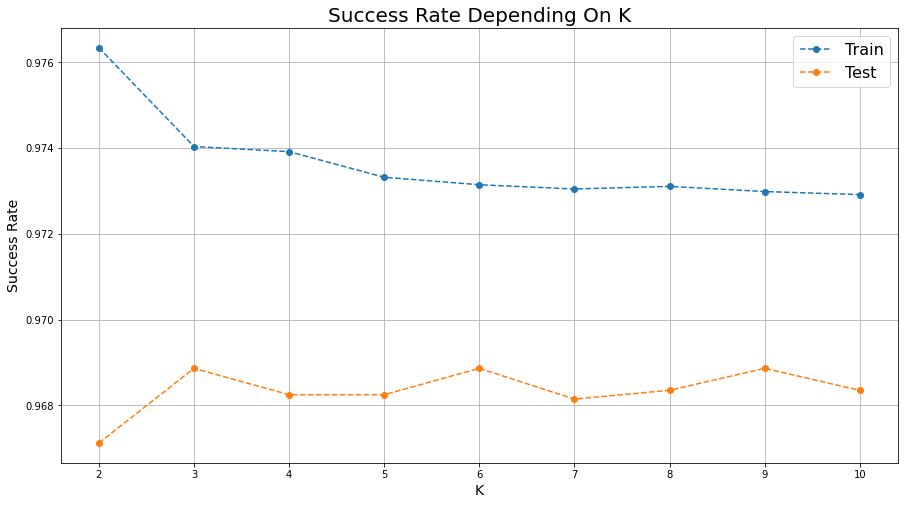
Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I

 - Test: Swizzor.gen!I

Chart, bar chart

Description automatically generated

****

**Train (Includes test results)**

K = 2

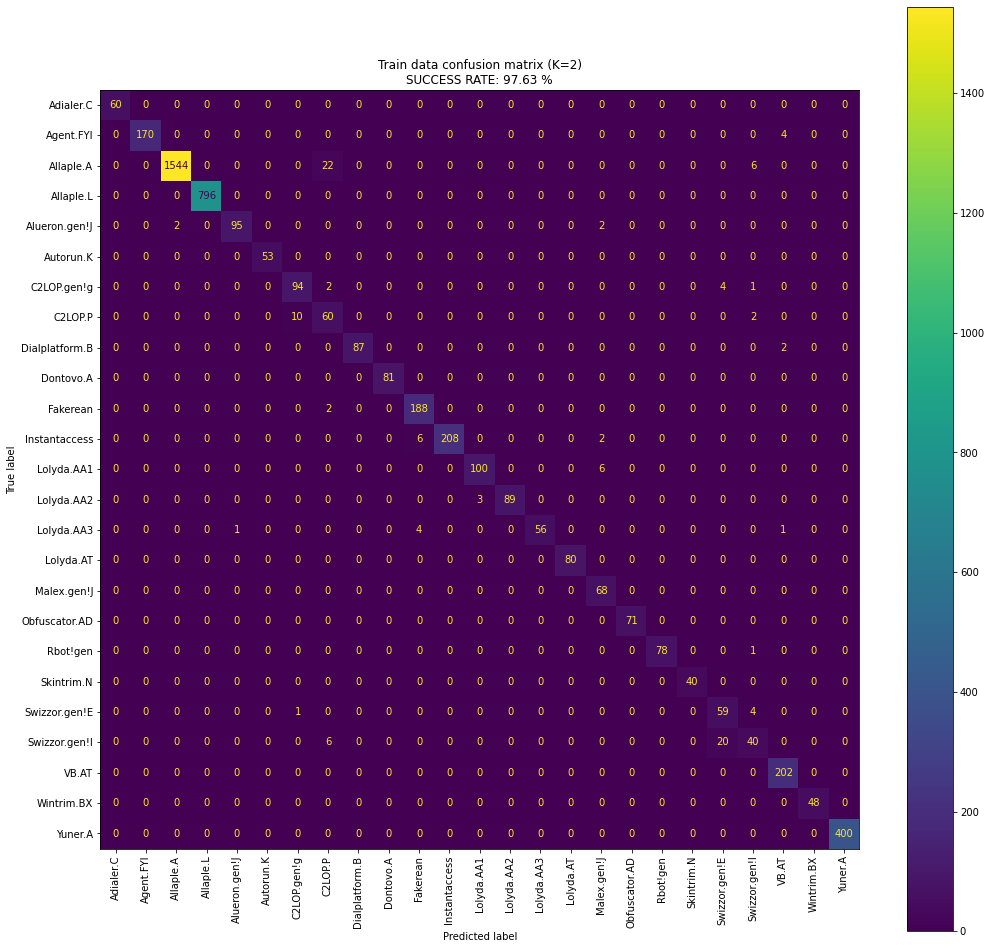
Success Rate (after normalizing):

- Train: 97.634 %

- Test: 96.712 %

Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I

**** - Test: Swizzor.gen!I

**Chart, background pattern, bar chart

Description automatically generated**

**Test (Includes train results)**

K = 3

Success Rate (after normalizing):

- Train: 97.404 %

- Test: 96.887 %

Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I

**A picture containing text, monitor, electronics, display

Description automatically generated** - Test: Swizzor.gen!I

**Chart, background pattern, bar chart

Description automatically generated**

* 1. CNN

**Train (Includes test results)**

טודו:

**Test (Includes train results)**

**Train (Includes test results)**

**Test (Includes train results)**

**Train (Includes test results)**

**Test (Includes train results)**

1. מסקנות

מחקר הפרויקט בא לבדוק האם שימוש בלמידה עמוקה באמצעות רשת CNN עדיף על פני שימוש בשיטות שונות מתחום למידת המכונה עבור בעיית סיווג תמונות ובפרט, תמונות של נוזקות.

* 1. מסקנות כלליות עבור מודלים פרימיטיביים
     1. LDA

מודל זה נתן את התוצאות הטובות ביותר מבין כל המודלים הפרימיטיביים עבור גדלי התמונות ו- , תוצאות של כ-97.009 וכ-97.347 אחוזי הצלחה בהתאמה. *אם כי כאשר רץ המודל על קלט של , ניתן היה להבחין בירידה דרסטית באחוזי ההצלחה – 70.7%.*

בנוסף, ניתן לראות שעבור גדלי התמונות ו- מירב השגיאות של המודל בסיווג הנוזקות אירעו עבור שתי משפחות עיקריות (Swizzor.gen!E, Swizzor.gen!I), אם נבחן את תוצאות מודלים אלו ללא התייחסות למשפחות אלה, נוכל להגיע לאחוזי דיוק טובים בהרבה.

לעומת זאת, כאשר אנו מסתכלים על המודל , ניתן לראות בעיה רוחבית בתהליך הסיווג *–*  לא רק עבור שתי המשפחות שציינו לעיל, אלא עבור מספר רב של משפחות. נוכל לפרק בעיה זו לשני חלקים. החלק הראשון של בעיה זו הוא בעיית סיווג התמונות עבור שתי המשפחותSwizzor.gen!E ו-Swizzor.gen!I, חלק זה של הבעיה קיים עבור כלל גדלי התמונות. חלקה השני של הבעיה מתרחש רק עבור גודל התמונות , חלק זה מעלה בעיה רוחבית בתהליך הסיווג. כלומר, קיימות מספר רב של משפחות אשר למודל יש קושי לזהותן *וניתן לומר כי בעיה זו נגרמת כתוצאה ממספר מאפיינים גדול יחסית למספר התמונות הנבדקות – על בעיה זו נרחיב בהמשך.*

היבט נוסף עליו רצוי לתת את הדעת הוא זמני ריצת המודל עבור כל אחד מהגדלים השונים. עבור אלגוריתם זה ניתן לראות זמני ריצה איטיים יחסית בהשוואה לשאר המודלים. דבר אשר יכול להוות מכשול כאשר מעוניינים לפתור בעיה זו בסביבה דלת אמצעיים.

* + 1. QDA

מודל זה נתן את התוצאות הגרועות ביותר מבין כל המודלים הפרימיטיביים עבור כלל גדלי התמונות.

כבר בשלבי ריצת המודל ניתן היה לשים לב לבעיה הנקראת Variable Collinear אשר מתרחשת כתוצאה מקורלציה חזקה מאוד בין שני משתנים שונים המקשה מאוד לבצע הערכה/סיווג על כל אחד מהם בנפרד. בעקבות בעיה זו, ניתן להסיק שהמודל אינו מתאים לפתרון הבעיה בה עוסק הפרויקט או לחילופין, מודל זה ידרוש התאמות רבות בניית המודל ועיצוב הנתונים. התאמות אלו אינן כלולות בסקופ הלימודי של פרויקט זה ועל כן לא נעסוק בכך.

* + 1. GNB
  1. משפחות בעלות דמיון
     1. כללי
     2. עבור מודלים ספציפיים
  2. זמני ריצה טודו: טבלאות של זמני ריצה וכמה מילים, זהו
  3. מסקנות עבור מודלים מסוג CNN

מסקנות כלליות

טודו: כללי פר מודל + ביחס למחקר

סיכום טודו: יש גרפים שעדן עשה באקסל

הפרויקט חקר את מאגר הנתונים Malimg בשלוש וריאציות שונות של גדלי תמונות: *, כאשר כל וריאציה היוותה קלט עבור כל אחד מהמודלים שנבדקו (*CNN, QDA, LDA, GNB*).*

*על סמך התוצאות שקיבלנו, בבדיקת סדרת המבחן עבור כל אחד מהמודלים, ניתן להסיק כי כל אחת מרשתות ה-*CNN *שהופעלו השיגה תוצאה טובה יותר מיתר המודלים (*QDA, LDA, GNB*) עבור אותה הקטגוריה.*

בנוסף, גם ביחס למאמר בתחום, עליו ביססנו את הפרויקט, ניתן לראות כי הצלחנו לשפר את כלל המודלים שנבדקו במספר אחוזים. למשל, עבור המחקר במאמר השיג כ-95.7% הצלחה ואילו מנגד, הצלחנו להשיג כ-99.7% אחוזי הצלחה. כאמור, כותבי המאמר מפרטים רק על העקרונות שעל בסיסם בנו את המודלים בניסוי, וכתוצאה מכך, נעזרנו בעקרונות אלה כקווים מנחים על מנת לנסות ולהתחקות לאופן פעולתם.

באשר למודלי למידת המכונה בהם השתמשנו, ניתן לראות בתוצאות שקיבלנו שעבור LDA, בקטגוריות של *, השגנו תוצאות גבוהות של כ-97% עבור כל אחד. אם כי כאשר רץ המודל על קלט של , ניתן היה להבחין בירידה דרסטית באחוזי ההצלחה – 70.7% בלבד.*

במודל ה-QDA ניתן להבחין כי כלל הקטגוריות נכשלו מבחינת אחוזי הצלחה כאשר כולם נמצאו מתחת ל-60%.

אנו מסיקים כי ירידה כה דרסטית באחוזי ההצלחה עבור כל המקרים הללו נובעת מהגדרה הנקראת

"Curse of Dimensionality". לפיה, מספר הדוגמאות שצריך על מנת שתהיה יכולת להעריך פונקציה שרירותית מסוימת בדיוק גבוה, גדל בצורה אקספוננציאלית ביחס למספר הקלטים אשר אותה פונקציה מקבלת.

עבור מודל ה-GNB, הצלחנו להשיג אחוזי הצלחה גבוהים יחסית, כ-96% עבור כל אחת מהקטגוריות. אנו סוברים כי בעקבות ההנחה של אי תלות סטטיסטית אשר מניח אלגוריתם ה-GNB, הוא מצליח להתגבר על הבעיה של קודמיו וזאת מכיוון שאי התלות מפחיתה את כמות המקרים שעל המודל ללמוד.