**מכללת אפקה: החוג למדעי המחשב**

דו"ח פרויקט גמר

זיהוי שטרות מקוריים

Fake banknotes detection

מגישים: אמיר פלג (205818032), אופק רודיך (209194414)

מנחה: ד"ר דינה גורן-בר

תאריך הגשה: 18/04/2023

**תודות**

ברצוננו להודות לד"ר דינה גורן-בר על ליווי צמוד והכוונה תוך כדי מתן אפשרות חופש פעולה, גמישות ומקום ליצירתיות במהלך העבודה על הפרויקט. תודה נוספת גם לעומר נחשון על הפידבקים, הזמינות הגבוהה, התמיכה והמסירות שהביאו להשלמת פרויקט זה.

**תוכן עניינים**

[תקציר מנהלים 1](#_Toc132663319)

[מבוא: 2](#_Toc132663320)

[מוטיבציה 2](#_Toc132663321)

[הגדרות הבעיה 2](#_Toc132663322)

[מטרות ויעדים 2](#_Toc132663323)

[סקירת ספרות 2](#_Toc132663324)

[סקר שוק 3](#_Toc132663325)

[חלופות 5](#_Toc132663326)

[ארכיטקטורה 8](#_Toc132663327)

[תכן 9](#_Toc132663328)

[תיאור התוצר 9](#_Toc132663329)

[אלגוריתמים 9](#_Toc132663330)

[קוד 10](#_Toc132663331)

[הדגמה 18](#_Toc132663332)

[הערכה 22](#_Toc132663333)

[Data Set 22](#_Toc132663334)

[מדדים 23](#_Toc132663335)

[צורת הבדיקה 26](#_Toc132663336)

[תוצאות 26](#_Toc132663337)

[סיכום ומסקנות 26](#_Toc132663338)

[רשימת מקורות 27](#_Toc132663339)

# תקציר מנהלים

הפרויקט עוסק בזיהוי של שטרות מזויפים באמצעות זיהוי תמונה.

חקרנו וגילינו שישנם שטרות מזויפים רבים בשוק ורצינו להעניק לאזרח הפרטי כלי פשוט, חינמי, ויעיל לזיהוי השטר בזמן אמת.

לאחר סקירת ספרות מעמיקה בנושא זיהוי תמונה בכלל וזיהוי שטרות בפרט, גילינו כי ישנן תכונות סטטיסטיות המבדילות בין שטרות מקוריים ובין שטרות מזויפים. במהלך העבודה, גילינו שחלק מתכונות אלו אינן משתקפות בתמונה המצטלמת במכשיר נייד, כמו חספוס ואיכות הנייר, אך תכונות שונות כגון צבעי השטר, ההדפס שעליו והמרחק הסטטיסטי בין הגוונים והניגודיות שלהם משתקפים בצורה מספקת דרך עדשת המכשיר הנייד.

לאחר מכן, ביצענו בדיקה של החלופות הקיימות כיום בשוק בנושא זיהוי השטרות המזויפים ונוכחנו לגלות שהיום אין יכולת לאדם הפרטי לעשות זאת בעצמו בצורה טכנולוגית, מכיוון שהמכשירים המוצעים היום בשוק הם לשימוש עסקי בלבד. במצב הנוכחי, זיהוי השטרות יכול להיעשות אך ורק באמצעות שינון כל הפרטים המזהים על השטר ובצורה ידנית ע"י אזרחים, ולא קיים כלי נגיש לזיהוי טכנולוגי וודאי יותר.

בחרנו בגישה המשלבת מספר אלגוריתמים מעולם למידת המכונה. תחילה, בחרנו במאגר נתונים חופשי לשימוש של דולרים המכיל פרטים סטטיסטיים על השטרות שבו. כפי שלמדנו בקורס למידת מכונה, וידאנו כי לא קיימים שדות ריקים במאגר הנתונים שעלולים לפגוע ביעילות ואמינות הקוד. על מאגר זה לימדנו ופיתחנו מודל המבוסס על רשתות נוירונים לטובת זיהוי שטרות עתידיים. לאחר מכן, עבור שטר חדש חילצנו את הנתונים הדרושים להכרעת מקוריות השטר. לבסוף, ביצענו השוואה של נתונים אלה אל מול המודל שפותח וקיבלנו החלטה (באחוזים) את רמת הסבירות שהשטר מקורי.

This project deals with counterfeit detection through image recognition.

We researched and discovered that there are many counterfeit bills on the market, and we wanted to give the common citizen a simple, free, and effective tool for detecting the bill in real time.

After an in-depth literature review on image recognition and banknote recognition, we discovered that there are statistical features that distinguish between original and counterfeit bills. During the work, we discovered that some of these features are not reflected in the image taken on a mobile device, such as the roughness and quality of the paper, but various features such as the colors of the banknote, the print on it, and the statistical distance between the shades and their contrast are adequately reflected through the lens of the mobile device.

After that, we carried out an examination of the alternatives that currently exist in the market regarding the detecting of counterfeit bills and we found out that today the common citizen does not have the ability to do this himself in a technological way, because the devices offered in the market today are for business use only. In the current situation, the identification of the banknotes can only be done by memorizing all the identifying details on the banknote and manually by citizens, and there is no accessible tool for more certain technological identification.

We chose an approach that combines several algorithms from the Machine Learning world. First, we chose a free-to-use database of dollars that contains statistical details about the bills in it. As we learned in Machine Learning course, we made sure that there are no empty fields in the database that could harm the efficiency and reliability of the model. On this database we taught and developed a model based on neural networks to identify future bills. After that, for a new banknote we extracted the necessary data to determine the banknote's authenticity. Finally, we compared this data against the developed model and determined (in percentages) the level of probability that the banknote is original.

# מבוא:

# מוטיבציה

כיום במדינת ישראל, 80% מהשטרות המזויפים מתגלים רק בעת ההפקדה בכספומט, כאשר רובם הינם שטרות של 200 ש"ח.

מחזור השימוש השנתי בשטרות מזומנים הוא כ115,000,000 ש"ח, מה שמהווה קרקע פורייה עבור מזייפי השטרות.

# הגדרות הבעיה

מרבית האנשים היום לא מכירים את כל הסממנים המזהים בשטרות, ואם אזרח מגלה כי קיבל שטר מזויף בעת ההפקדה בכספומט, הרי שגילה זאת מאוחר מדי, ומוטב היה לגלות זאת בעת קבלה השטר לאחר בדיקה מהירה ויסודית. אנו מודעים לעובדה שברוב העסקים הקטנים, ובטח שלאדם הפרטי, לא קיים כלי מתוחכם לבדיקת מקוריות השטר.

# מטרות ויעדים

ברצוננו ליצור אלגוריתם המאמת את מקוריות השטר באחוזים.

אנו מעוניינים לעשות זאת במהירות, תוך דיוק של לפחות 80%, בהתאם למגבלות הידועות בטכנולוגית הטלפונים הסלולריים והמגבלה שבסל הנתונים (Data set) שתפורט בהמשך.

# סקירת ספרות

לאור עלייה בזמינותם ויכולת העיבוד הגוברת של המכשירים הסלולריים בקרב הציבור ניתן כיום לבצע חישובים שהיו נחשבים בעבר כחישובים מורכבים מכל מקום בפשטות ובנוחות, מעבר ליכולת לביצוע חישובים מורכבים לטובת זיהוי האם השטר הוא שטר אמיתי או מזויף ניתן להשתמש במכשירים אלו לטובת סיוע לאנשים בעלי לקויות ראייה בזיהוי השטרות. מחקרים שונים שבוצעו הראו כי שטרות מזויפים נוטים להכיל תכונות שונות משטרות אמיתיים עקב הקושי ביצירת שטר מזויף הזהה במרכיביו ובתכונותיו לשטר מקורי, בחרנו מחקר ספציפי[1] שציין כי ניתן לראות הבדלים אלו באופן ניכר בשטר באזורים מסוימים, אזור המספר הסידורי (Serial Number) ודגל המדינה (Flag) הם האזורים שנחשבים לאזורים "קשים" יותר לזיוף ולכן באמצעות מיקוד, סריקה ובדיקה של אזורים אלו ניתן להגיע לאחוזי דיוק גבוהים יותר

כיום ישנם מספר מחקרים שמציגים אלגוריתמים לביצוע תהליך זיהוי השטרות, רובם ככולם מתחילים בתהליך זהה של צילום השטר וביצוע השוואות שונות אל מול Dataset שמכיל מידע ותמונות בנוגע לשטרות מקוריים ומזויפים או לחליפין מכיל מידע אודות תכונות שהופקו מDataset, לאחר ביצוע הצילום מתבצעות פעולות מתמטיות וחישובים על גבי התמונה לצורך פירוק וחילוץ תכונות "קריטיות" המשמשות לזיהוי וקטלוג השטר, מחקר אחר מציין כי על מנת לזהות את השטר עליו לבצע צילום של השטר מזוויות שונות על מנת ליצור מצב של תמונת "360" בשביל לתת זיהוי וודאי באיזה שטר מדובר ומהו סכומו.[2] תהליך הפירוק אינו בהכרח זהה במחקרים השונים ותלוי באופן ישיר באופן החישוב שנלווה לתהליך. אנו בחרנו להתמקד ב2 גישות מרכזיות בהם המחקרים מתעמקים וזה בהתאם למשאבים ושלבים מתקדמים יותר בתהליך הזיהוי.

התכונות הקריטיות בהן המחקרים מתמקדים הן חומר הנייר (השטר), סוג וצבע הדיו שמשתנה בין השטרות ואף בנקודות שונות על גבי אותו השטר, איכות הנייר, חספוס פני השטח ומידות השטר[3], כפי שצוין הבחירה במאפיינים אלו רלוונטית ותלויה באופן בו ננתח את המידע ובדיקתו מול מאגר המידע שנאסף עבור אוסף השטרות. אופן השגת תכונות אלו בא לידי ביטוי בטכניקות פירוק והפקה של תמונות ברזולוציות גבוהה כגוןScanning electron microscopy (SEM) וX-ray Diffraction (XRD) המספקות אינפורמציה רבה על סוג, איכות הדיו, חומר השטר וחספוסו.

# סקר שוק

בעבודתנו סקרנו מספר פתרונות לבדיקת מקוריות השטר הקיימים כיום בשוק:

1. כספומט

כספומט בבנקים ברחבי ישראל (או כספומטים פרטיים), דרכם ניתן להפקיד ולמשוך מזומן ופעולות נוספות.

מידע נוסף [4] על הטכנולוגיה

1. מכונה לממכר מזון

מכונה באמצעותה ניתן לרכוש מזון ושתיה באמצעות מטבעות, כרטיס אשראי ולעיתים שטרות.

דוגמה למוצר [5] מהטכנולוגיה

1. גלאי שטרות - Banknote Detector

מכשיר לאימות השטר, לעיתים גם סופר את השטרות וסוכם את ערך השטרות שהוכנס אליו.

דוגמה למוצר [6] מהטכנולוגיה

1. יישומי סלולאר קיימים

יישומים במכשיר הנייד, אשר מזהים את ערך השטר ובודקים האם השטר אמיתי.

דוגמאות ליישומי סלולאר [7][8]

**תובנות:**

כיום ישנם מכשירים איכותיים אשר יודעים לאמת את אותנטיות השטר כדוגמת כספומטים, מכונות שתיה וכו' - מכשירים מסיביים ויקרים, עם בדיקות של כ- 100% דיוק באמצעות בדיקות עובי, פס מגנטי ובדיקות רבות נוספות, אשר אדם פרטי לא יכול לרכוש ובדיקת אותנטיות השטרות אינה מטרתם היחידה.

גלאיי שטרות- אלו מכשירים שכל תפקידם הוא לאמת את מקוריות השטר, גם הם בעלי בדיקות רבות כמו פס מגנטי, אינפרה אדום וכו' ובעלי 100% הצלחה, אך מחירם יקר מאוד גם כן.

אנו רוצים להתמקד בבדיקת השטר באמצעות מצלמת הטלפון החכם, שכידוע לא יוכל להגיע לרמת דיוק זהה למכשירים לעיל, אך יכול לשמש כל אדם כיוון שכיום כ- 91% מאוכלוסיית העולם מחזיקה במכשיר חכם המסוגל לעשות זאת.

לכן בדקנו את היישומים העושים זאת כיום- הממצאים הם כי אין יישומים היודעים לקבוע האם שטר מזויף או לא, אלא רק להגיד מהו ערך השטר. אלו היודעים לקבוע שאינו אמיתי, ידעו רק במקרים בהם השטר רחוק מאוד מלהידמות לשטר אמיתי (שטרות מונופול למיניהם). אנחנו רוצים להגיע לרמה הפרדה של שטרות קרובה יחסית לשטרות אמיתיים ושלא כל אדם יכול במבט ראשון לדעת כי השטר מזויף.

אנו מודעים למגבלות הטכנולוגיה של הטלפונים הניידים כיום, ולא מתיימרים להתחרות בכספומטים והמכשירים הדומים.

לטענתנו, רוב האנשים לא מכירים את כל הסממנים הוויזואליים שיש לבדוק בשטר כדי להוכיח שהוא אמיתי. אנו מעוניינים שהאלגוריתם שלנו הוא שיעשה את הבדיקה הזו, ולכן השילוב של בדיקת אותנטיות ויזואלית של השטר ומצלמת הטלפון הנייד, הוא פתרון שכיום לא קיים בשוק כפי שניתן לראות.

**טבלת השוואה:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **כספומט** | **מכונות לממכר מזון** | **גלאי שטרות** | **אפליקציות קיימות** | **הפתרון שלנו** |
| **ייעוד** | *הפקדת כספים / משיכת כספים ועוד* | *רכישת מזון ממכונה* | *בדיקת שטרות (חלקם גם סופרים שטרות)* | *הקראת ערך השטר* | *בדיקת שטרות* |
| **האם מזהה שטרות מזויפים** | כן | כן | כן | לא | כן |
| **מחיר** | לא זמין לרכישה | אלפי שקלים | מאות שקלים עד אלפי שקלים | חינמי | חינמי |
| **טכנולוגיה בזיהוי השטרות** | זיהוי גודל תלת מימד, מצלמה, אינפרה אדום, אולטרה סגול, חוט מגנטי, סימן מים | רובן לא מקבלות אך אלו שכן- חיישנים אופטיים ומגנטיים | זיהוי גודל תלת מימד, חוט מגנטי, אינפרה אדום, דיו מגנטי, סימן מים | מצלמה | מצלמה |
| **נייד/נייח** | נייח | נייח | נייד | נייד | נייד |
| **זמינות** | ישנם בודדים בשכונה שלמה | נמצא רק במקומות ציבוריים | רק מי שיש לו את המכשיר | לכל אדם עם מכשיר נייד, 24/7 | לכל אדם עם מכשיר נייד, 24/7 |
| **כמות שטרות בפעולה** | עד 50 שטרות | עד 5 שטרות | שטר אחד | שטר אחד | שטר אחד |

# חלופות

**עצי החלטה - Decisions trees**

עץ החלטות יעבוד על ניתוח קבוצה של תכונות הקשורות לשטרות הנמצאים במאגר המידע על מנת ליצור מודל שיוכל להבדיל ביניהם, עץ ההחלטות "ילמד" את נתוני המאגר על מנת ליצור מערכת כללים לסיווג שטרות חדשים, בכל צומת של העץ האלגוריתם בוחר את התכונה המספקת את "הרווח" הגדול ביותר בהקשר של מידע. העץ ימשיך לפצל את הנתונים עד שהמחלקות (מזויף/מקורי) יהיה מופרדות ככל האפשר, לאחר בניית ולמידה של העץ ניתן לבחון עליו שטרות חדשים ולקבל תשובה האם הם מקוריים או לא

יתרונות:

1. לעצי החלטה יש יתרון מובהק כאשר משתמשים בו במודלים הכוללים מספרים וחישובים מתמטיים. העצים קלים להבנה וניתן להמחיש בקלות את הכללים המשמשים לסיווג ולהעביר אותם לבעלי עניין, מה שהופך את עצי ההחלטה לכלי שימושי להעברת תוצאות גילוי הזיופים ואופן הבדיקה לקהלים שאינם טכניים.
2. חישוב עצי החלטה הוא יעיל מבחינה חישובים ומאפשר ניתוח של מערכי נתונים גדולים בפרק זמן יחסית מהיר. מה שהופך אותם לשימושיים לעיבוד כמויות גדולות של נתונים במהירות, מה שחשוב בהקשר של זיהוי שטרות שבו עשוי להיות נפח גדול של פריטים לנתח.

חסרונות:

1. כפי שניתן לראות בהמשך עבור Entropy וKurtosis, חלוקת הנתונים איננה אחידה ולכן יתכן והעץ יהיה מוטה עבור ערכים גבוהים יותר (מכיוון שהרווח שלהם גדול יותר) ובכך לעוות את התוצאה הרצויה.
2. עצי החלטה רגישים לשינויים קטנים בהרכב הקבוצות אותן הם לומדות ויכולים לייצר עצים שונים עבור תתי קבוצות שונות של אותם הנתונים, המשמעות היא שהמודל עשוי להיות לא עקבי בשימושים חוזרים עם אותו מאגר מידע.

**אלגוריתם שכן קרוב – k-Nearest Neighbors algorithm (k-NN)**

אלגוריתם זה משמש לבעיות סיווג ורגרסיה על ידי השוואת "מרחקים" בין שדות שונים (תכונות) במרחב המידע, על מנת להשתמש באלגוריתם זה יש לעבד מראש את מערך הנתונים על ידי סטנדרטיזציה או נורמליזציה של התכונות כדי להבטיח שהן באותו קנה מידה. האלגוריתם מאחסן את כל מערך הנתונים ומקצה כל שטר לK שכניו הקרובים ביותר במרחב התכונות. כאשר נכניס לאלגוריתם שטר חדש, האלגוריתם ימצא את K שכניו הקרובים ביותר ובהתאם לכך ידע לקטלג את השטר החדש

יתרונות:

1. המימוש וההבנה של אלגוריתם k-NN הוא יחסית פשוט ליישום ומובן להבנה, עבור פרויקט ודריסת רגל ראשונה בתחום למידת המכונה יכול להקל על התהליך.
2. האלגוריתם אינו דורש שלב של Train ולכן מאפשר לו לפעול באופן מהיר מאוד בזמן אמת בלי פגיעה בתוצאה הרצויה ולכן ניתן "לבנות" את המודל ולקבל את התוצאות הרצויות בזמן קצר.

חסרונות:

1. סיבוכיות האלגוריתם היא גבוהה, מכיוון n שטרות, עבור כל שטר נצטרך לבצע את בדיקת המרחק (d) עבור כל K שכניו הקרובים ונקבל סיבוכיות של O(knd).
2. החשיבות של בחירת מספר ה-K היא גבוהה מאוד, על ידי בחירת ערך K שגוי (גבוה/נמוך מדי) יתכן ונקבל תוצאות שגויות (התאמת יתר או תת התאמה בהתאם לערך הK), ולכן לבחירת הערך יש חשיבות גבוהה ביעילות ואמינות האלגוריתם.

**אלגוריתם k - מרכזים – (K-Means algorithm)**

מטרת האלגוריתם היא לחלק את התצפיות (הנתונים עבור כל שטר) ל-k אשכולות לפי מרכזי כובד (k-means). כל תצפית משויכת לאחד מ"מרכזי הכובד". על ידי בחירה נכונה של מרכזי כובד ניתן לאתר את הקבוצות השונות. זהו מודל סטטיסטי שאינו מתבסס על ידע מוקדם על הנתונים אלא רק על תצפיות בפועל.

יתרונות:

1. אלגוריתם k – מרכזים אינו דורש נתונים מקוטלגים לצורך האימון ולכן הופך אותו לפופולארי לשימוש בשיטת Unsupervised learning או לחליפין לזיהוי תמונות או טקסט.
2. ניתן לשפר באופן יחסית פשוט את דיוק האלגוריתם על ידי הרצות נוספות שלו עבור k משקלים שונים ולבצע אינטגרציה עם המשקלים שנוצרו עד כה.

חסרונות:

1. בדומה לאלגוריתם k-NN גם כאן יש חשיבות גבוהה עבור ערך ה-K שנבחר, עבור בחירת ערך K שגויה המודל יפיק לנו תוצאות שגויות ותוך כדי התייחסות ליתרון מספר 2 (שיפור הדיוק) חסרון זה יכול לבלוט באופן משמעותי.
2. במידה ונבחר להשתמש באלגוריתם עם מאגר מידע מוכן דרוש עיבוד מקדים של המאגר תוך כדי הסרה/תיקון של ערכים ריקים ונרמול התכונות, עיבוד זה יכול לדרוש זמן ודורש היכרות מוקדמת עם הנתונים וידע בתחום.

**ניתוח גורמים ראשיים – Principal Component Analysis (PCA)**

טכניקה פופולארית לטובת עיבוד מקדים של המידע בכך שהיא מפחיתה את ממדי הקלט לממד נמוך יותר טוב שמירה על מידע חשוב מהנתונים המקוריים, בנוסף טכניקה זו עוזרת לחילוץ מאפיינים רבים מן הקלט על מנת לבנות את המודל או להשוות קלט עתידי אל מול המודל לקבלת תוצאה

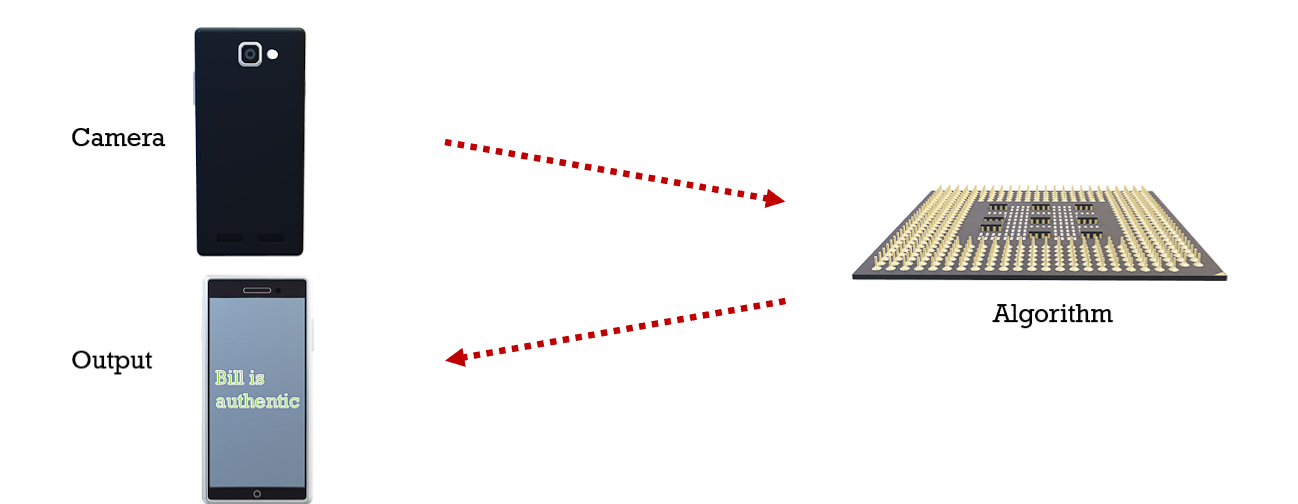
יתרונות:

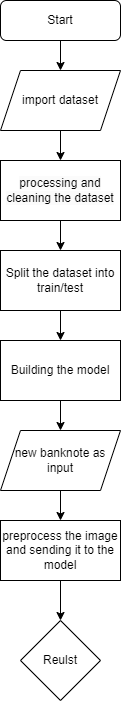
1. שימושה העיקרי והבולט ביותר של טכניקה זו היא היכולת להפחית במדי הקלט ובכך להקל על יכולת הניתוח ובניית מודל תוך שמירה על תכונות ומאפיינים הכרחיים והסרה או התעלמות מתכונות אחרות
2. זיהוי וחילוץ של התכונות והמאפיינים החשובים עבור קלטים לטובת בניית מודל מיומן

חסרונות:

1. הסיכון בשימוש בטכניקה זו היא אובדן של מידע, יתכן ומידע יוגדר מסיבה כלשהי כמידע "לא חשוב" אך בפועל מדובר במאפיינים הכרחיים לטובת מימוש יעיל וזיהוי מוצלח
2. בדומה לאלגוריתמים קודמים, גם כאן יש חשיבות לגודל הממד הרצוי אותו נקבל, בחירה של מימד שאיננו נכון עלולה לגרום לאובדן מידע ולחוסר אמינות במודל

# ארכיטקטורה



****

טרם נתחיל בפירוט השלבים באלגוריתם עלינו לציין (ויפורט על כך בהמשך) שהארכיטקטורה שלנו שונתה במקצת עקב שינוי בDataset איתו עבדנו.

1. import dataset – השלב הראשוני, ייבוא מאגר הנתונים איתו נרצה לפתח את המודל.
2. processing and cleaning the dataset – עקב הבחירה בשימוש באלגוריתמים קיימים מסוג K-Means וPCA עלינו לוודא כי הDataset "נקי", שלא קיימים שדות ריקים ושהמידע הרלוונטי מוגדר ככזה על מנת שלא יאבד במהלך התהליך.
3. Split the dataset into train/test – חלוקת המידע שלנו למחלקות Train וTest על מנת לפתח את המודל.
4. Building the model – בניית המודל על סמך מאגר המידע הנקי והעדכני לאחר שלבים 1-3

**שלבים 1-4 מתבצעים באופן חד פעמי לצורך יצירת המודל, על מנת לתחזק את המודל ולעדכנו יהיה צורך לתקף שלבים אלו עבור כל שינוי.**

1. New banknote as input – כעת לאחר שקיים לנו מודל מיומן נרצה לקבל תוצאות עבור הקלט שלנו, בשלב זה "נכניס" לקוד את התמונה של השטר אותו נרצה לבדוק
2. preprocess the image – טרם שליחת התמונה למודל נרצה לבצע עליו פעולות מקדימות, בשלב זה נקטין את התמונה לגודל 64x64 פיקסלים, המרת התמונה לגווני אפור (Grey Scale), שיטוח התמונה למערך חד ממדי ונרמול הערכים.
3. Decision – לאחר שבוצעו כל השלבים הנ"ל התמונה תשלח אל המודל אשר תפקידו לחזות את המחלקה של הקלט, אם המחלקה שהתקבלה מהמודל הינה קטנה או שווה ל0.5 השטר נחשב כמזויף (מחלקה 0), אם הוא גדול מ0.5 השטר יחשב כשטר אמיתי (מחלקה 1).

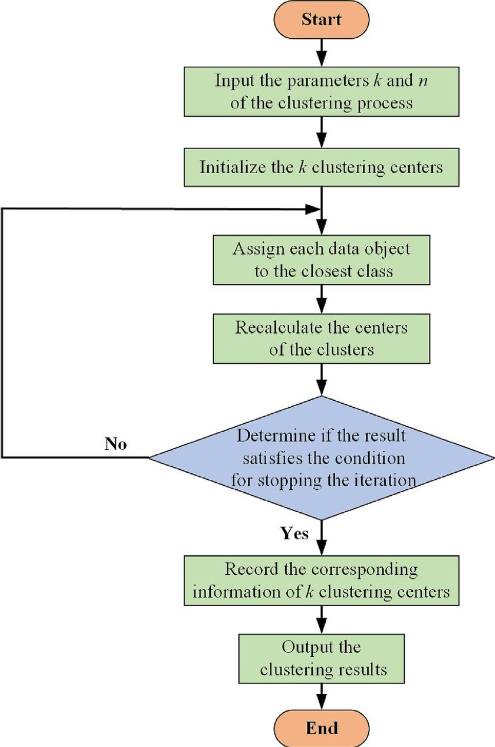
# תכן

במהלך הפרויקט בוצעו כמה שינויים בהתאם לאילוצים ובכדי להביא לשיפור הזיהוי שבפרויקט.  
כאמור, מאגר הנתונים שונה ולכן זיהוי התמונה לא מתבסס על ROI ועל חספוס אלא על המדדים שצוינו.

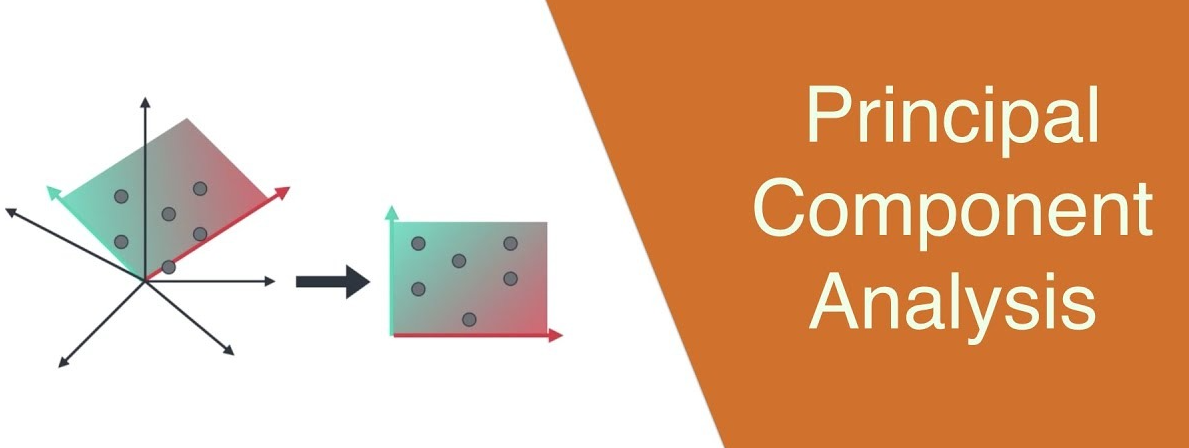
תחילה, רצינו לעבוד עם YOLOv5, אך לאחר מכן החלטנו לעבור לCNN. בנוסף להיותו גמיש יותר, הסיבה המרכזית לשינוי היא שעבודה עם רשתות נוירונים דורשת כוח חישובי נמוך יותר מYOLOv5, יתרון משמעותי כאשר מדובר בטלפונים ניידים.

כפי שיפורט בהרחבה בסעיף הבא בחרנו להשתמש במספר של אלגוריתמים על מנת להמחיש את המידע הקיים במאגר הנתונים ועל מנת לבנות מודל שידע לקבוע עבור קלטים עתידיים את המחלקה שלהם (שטר מזויף/שטר מקורי)

**אלגוריתם k - מרכזים – (K-Means algorithm)**



**ניתוח גורמים ראשיים – Principal Component Analysis (PCA)**



**אלגוריתם CNN (Convolutional Neural Network)**

Diagram

Description automatically generated

# תיאור התוצר

# אלגוריתמים

**אלגוריתם k - מרכזים – (K-Means algorithm)**

מטרת האלגוריתם היא לחלק את התצפיות (הנתונים עבור כל שטר) ל-k אשכולות לפי מרכזי כובד (k-means). כל תצפית משויכת לאחד מ"מרכזי הכובד". על ידי בחירה נכונה של מרכזי כובד ניתן לאתר את הקבוצות השונות. זהו מודל סטטיסטי שאינו מתבסס על ידע מוקדם על הנתונים אלא רק על תצפיות בפועל. בנוסף, אלגוריתם זה אינו דורש נתונים מקוטלגים לצורך האימון ויתרון זה הופך אותו לנפוץ בלמידת מכונה.

**ניתוח גורמים ראשיים – Principal Component Analysis (PCA)**

טכניקה פופולארית לטובת עיבוד מקדים של המידע בכך שהיא מפחיתה את ממדי הקלט לממד נמוך יותר תוך שמירה על מידע חשוב מהנתונים המקוריים, בנוסף טכניקה זו עוזרת לחילוץ מאפיינים רבים מן הקלט על מנת לבנות את המודל או להשוות קלט עתידי אל מול המודל לקבלת תוצאה

**אלגוריתם Adam (Adaptive Moment Estimation)**

אדם הוא אלגוריתם אופטימיזציה המשמש לאימון רשתות עצביות עמוקות בלמידת מכונה. הוא מתאים באופן אדפטיבי את קצב הלמידה על סמך שיפוע הפרמטרים ומהווה שיטת אופטימיזציה יעילה מבחינה חישובית ונפוצה.

**אלגוריתם CNN (Convolutional Neural Network)**

רשת עצבית עמוקה המשמשת בדרך כלל עבור זיהוי תמונות וסרטים. אלגוריתם זה משתמש בשכבות כינוס (convolution layers) על מנת לחלץ תכונות מתוך הקלט באופן אוטומטי, לאחר החילוץ תכונות אלו משמשות את המודל לסווג את התמונות למחלקות השונות.

# קוד

*#opening the DataFrame* **import** pandas **as** pd

df = pd.read\_csv('BankNote\_Authentication.csv') display(df)

classes = df['class'] features = df.drop('class', axis=1) **import** visuals **as** vs

%load\_ext autoreload

%autoreload 2

on the next cells we will illustrate our dataset and his features, it will help to understand the data distribution

vs.distribution(df) vs.scatter(df)

*#dimension of the DataFrame*

print("Number of rows: {}".format(df.shape[0])) print("Number of columns: {}\n".format(df.shape[1]))

n\_records = len(df)

n\_fake\_notes = len(df[classes == 0]) n\_real\_notes = len(df[classes == 1])

print("Total number of records: {}".format(n\_records)) print("Total number of fake notes: {}".format(n\_fake\_notes)) print("Total number of real notes: {}".format(n\_real\_notes))

*#counting the number of missing values in each column* missing\_values = df.isnull().sum().sum() **if** missing\_values == 0: print("\nThere are no missing values in the dataset") **else**: print("\nThe dataset has {} missing values".format(missing\_values))

print(df.isnull().sum())

*#Index object representing the column labels of the DataFrame* df.columns

*#Showing information about the DataFrame* display(df.describe())

*#setting maplotlib to show plots in the notebook and ingoring warnings* **import** matplotlib.pyplot **as** plt

%matplotlib inline **import** warnings

warnings.filterwarnings('ignore')

Here we illustrate the division of Variance v Curtosis since it is very easy to understand and to see the difference between them colors = {'0': 'red', '1': 'green'}

plt.scatter(df.variance, df.curtosis, alpha=0.5, c=df['class'].apply(**lambda** x: colors[str(x)]))

plt.title('Scatter Plot of Variance v Curtosis') plt.xlabel('Variance') plt.ylabel('Curtosis') plt.show()

There is no obvious cluster in spherical shapes. We had to check many K's values but we found that 9 is the optimal as we can see in the next cell

*# k-mean implementation on dataset with a loop of iterations to check whether k mean is stable or not*

**import** numpy **as** np **from** sklearn.cluster **import** KMeans

n\_iter = 9

fig, ax = plt.subplots(3, 3, figsize=(10,10)) ax = np.ravel(ax) **for** i **in** range(n\_iter): km = KMeans(n\_clusters=2,max\_iter=3) km.fit(df)

centroids=km.cluster\_centers\_

ax[i].scatter(df[km.labels\_== 0]['variance'], df[km.labels\_ == 0]

['skewness'],label='cluster 1')

ax[i].scatter(df[km.labels\_ == 1]['variance'], df[km.labels\_ == 1]

['skewness'],label='cluster 2')

ax[i].scatter(centroids[:, 0], centroids[:, 1],c='r', marker='\*', s=100, label='centroid') ax[i].legend() plt.tight\_layout();

After running K-Means 9 times, the results we got are very similar, which means the KMeans is stable.

*# cloning df into df1 and keeping only 2 feature*

df1=df.copy()

df1.drop(['curtosis','entropy','class'],axis=1,inplace=True) df1.head()

Here we illustrate the division of Variance v Skewness since it is very easy to understand and to see the difference between them

clusters = KMeans(2)

clusters.fit(df1)

df1['clusterid'] = clusters.labels\_ colors = {'0': 'red', '1': 'green'}

plt.scatter(df1.variance, df1.skewness, alpha=0.5, c=df1['clusterid'].apply(**lambda** x: colors[str(x)]))

plt.title('Scatter Plot of Variance v Skewness') plt.xlabel('Variance') plt.ylabel('Skewness') plt.show()

*#getting centroids of cluster* clusters.cluster\_centers\_ df1.head()

*# calculating descriptive statistics for each cluster* df1.groupby( 'clusterid' ).describe() **from** sklearn.preprocessing **import** StandardScaler

*#normalization the data, 0 mean and 1 variance* scaler = StandardScaler()

scaled\_df1 = scaler.fit\_transform( df1[["variance", "skewness"]] )

scaled\_df1=pd.DataFrame(scaled\_df1,columns=['variance','skewness']) scaled\_df1

*# reproducibility of results and switching ID's*

clusters\_new = KMeans( 2, random\_state=42 ) clusters\_new.fit(scaled\_df1)

df1["clusterid\_new"] = clusters\_new.labels\_ df1.head()

colors = {'0': 'red', '1': 'green'}

*#plt.scatter(df1.variance, df1.skewness, alpha=0.5, c=df1['clusterid'].apply(lambda x: colors[str(x)]))*

plt.scatter(df1.variance,df1.skewness,alpha=0.5, c=df1['clusterid\_new'].apply(**lambda** x: colors[str(x)])) plt.show()

*#visualizing the data with correct labels*

plt.scatter(df['variance'],df['skewness'],c=df['class']) plt.xlabel('Variance')

plt.ylabel('Skewness') plt.colorbar(label='class') plt.show()

*# new centroids of clusters* clusters\_new.cluster\_centers\_

df1["clusterid\_new"] = df1["clusterid\_new"].map({0: 1, 1: 0}) df1.shape

df1=df1.reset\_index() df1

*# evaluate the accuracy of a clustering algorithm that assigns cluster labels*

correct=0 **for** i **in** range(0,1371): **if** df['class'][i]==df1['clusterid\_new'][i]: correct=correct+1 print(correct/1372)

Using PCA model to reduce the number of features into 2 columns df.head()

*#normalization the data, 0 mean and 1 variance* **from** sklearn.preprocessing **import** StandardScaler

scaler = StandardScaler()

scaled\_df2 = scaler.fit\_transform( df[["variance",

"skewness","curtosis","entropy"]])

scaled\_df2=pd.DataFrame(scaled\_df2,columns=['variance','skewness',"cur tosis","entropy"]) scaled\_df2

*# reducing dimensionality* **from** sklearn.decomposition **import** PCA

pca=PCA(n\_components=2) pca.fit(scaled\_df2)

PCA\_df=pd.DataFrame(pca.transform(scaled\_df2),columns=(['col1','col2']

))

PCA\_df.head()

plt.figure(figsize=(5,5))

plt.scatter(PCA\_df['col1'],PCA\_df['col2'])

plt.title('Scatter Plot of col1 vs col2') plt.show()

*#determine the optimal number of clusters for KMeans*

cluster\_range=range(1,11) cluster\_errors=[] **for** num\_clusters **in** cluster\_range: clusters=KMeans(num\_clusters) clusters.fit(PCA\_df)

cluster\_errors.append(clusters.inertia\_) plt.figure(figsize=(6,4))

plt.plot(cluster\_range,cluster\_errors,marker='o') plt.title('Elbow method to find the number of clusters') plt.show()

*# performing hierarchical clustering on a dataset represented by the PCA transformed dataframe PCA\_df.* **from** sklearn.cluster **import** AgglomerativeClustering

AC=AgglomerativeClustering(n\_clusters=4) AC.fit(PCA\_df)

yhat\_AC=AC.fit\_predict(PCA\_df) yhat\_AC

PCA\_df['Clusters']=yhat\_AC PCA\_df

plt.scatter(PCA\_df['col1'],PCA\_df['col2'],c=PCA\_df['Clusters']) plt.show()

Training the Model with the dataset

**import** pandas **as** pd **import** numpy **as** np

**from** sklearn.model\_selection **import** train\_test\_split **from** keras.models **import** Sequential **from** keras.layers **import** Dense, Dropout **from** keras.optimizers **import** Adam **from** keras.callbacks **import** ModelCheckpoint **from** keras **import** regularizers model = Sequential()

*# load dataset* df = pd.read\_csv('BankNote\_Authentication.csv') *# split data into training and testing sets* X\_train, X\_test, y\_train, y\_test = train\_test\_split(df.drop('class',axis=1), df['class'], test\_size=0.2)

*# build the model* model = Sequential()

model.add(Dense(128, activation='relu', input\_dim=X\_train.shape[1])) model.add(Dropout(0.2))

model.add(Dense(64, input\_dim=64, activation='relu', kernel\_regularizer=regularizers.l2(0.01))) model.add(Dropout(0.2)) model.add(Dense(1, activation='sigmoid'))

*# compile the model* optimizer = Adam(lr=0.001)

model.compile(loss='binary\_crossentropy', optimizer=optimizer, metrics=['accuracy'])

*# set up a checkpoint to save the best model during training* checkpoint = ModelCheckpoint('model.h5', monitor='val\_accuracy', save\_best\_only=True, mode='max')

*# train the model*

history = model.fit(X\_train, y\_train, epochs=50, batch\_size=300, validation\_data=(X\_test, y\_test), callbacks=[checkpoint], verbose = 0)

*# evaluate the model on the testing set*

loss, accuracy = model.evaluate(X\_test, y\_test, verbose = 0)

*# save the final model* model.save('model.h5') print("Model saved into model.h5 file")

**def** detect\_image(predicted\_class,img):

*# Load image*  image = cv2.imread(img)

*# Check if image has been loaded correctly*  **if** image **is** None: print("Error: Could not load image") **return**

*# Convert image to grayscale*

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY) gray\_blur = cv2.GaussianBlur(gray, (21, 21), 0) fig, ax = plt.subplots(figsize=(10, 10))

heatmap = ax.imshow(gray\_blur, cmap='inferno') fig.colorbar(heatmap, ax=ax)

*# Display the heatmap*  plt.show()

*# Apply adaptive thresholding to get binary image*  thresh = cv2.adaptiveThreshold(gray, 255, cv2.ADAPTIVE\_THRESH\_MEAN\_C, cv2.THRESH\_BINARY\_INV, 11, 2)

*# Find contours in the binary image*  ret, thresh = cv2.threshold(gray, 127, 255, 0)

contours, hierarchy = cv2.findContours(thresh, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

*#contours, hierarchy = cv2.findContours(thresh, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)*

*# Draw contours on original image*  *# Calculate spread of largest contour*  max\_contour = max(contours, key=cv2.contourArea) x, y, w, h = cv2.boundingRect(max\_contour) spread = w / h

*# Draw contours on original image*  img\_copy = image.copy() **if** predicted\_class<=0.5:

colors\_tuple = (0, 0, 255) **else**: colors\_tuple = (0, 255, 0)

cv2.drawContours(img\_copy, contours, -1, colors\_tuple, 2) print("Number of contours found:", len(contours)) print("Spread of largest contour:", spread)

*# Display the image with detected contours*  cv2.namedWindow("Detected Contours", cv2.WINDOW\_NORMAL) cv2.resizeWindow("Detected Contours", img\_copy.shape[1], img\_copy.shape[0])

cv2.imshow("Detected Contours", img\_copy) cv2.waitKey(0) cv2.destroyAllWindows()

**import** cv2 **import** numpy **as** np

**from** keras.models **import** load\_model

**from** sklearn.neighbors **import** KNeighborsClassifier **from** scipy.stats **import** \* **from** keras.models **import** load\_model **from** tensorflow **import** keras

**import** h5py

**import** matplotlib.pyplot **as** plt

**def** preprocess\_image(image): *# Load the trained model*  model = load\_model('model.h5', compile=False)

*# Resize the image to 64x64 pixels*  resized\_image = cv2.resize(image, (64, 64))

*# Convert the image to grayscale*

gray\_image = cv2.cvtColor(resized\_image, cv2.COLOR\_RGB2GRAY)

*# Flatten the image to a 1D array*  flattened\_image = gray\_image.flatten()

*# Normalize the image by dividing each pixel value by 255*  normalized\_image = flattened\_image / 255.0

*# Add two extra zeros to the end of the flattened image to make it a 1D array of four features*

feature\_vector = np.concatenate([normalized\_image, [0, 0]])

var = np.var(flattened\_image)/1372 skewness = skew(flattened\_image.reshape(-1)) kurt = kurtosis(flattened\_image.reshape(-1)) hist, \_ = np.histogram(flattened\_image, bins=256) entropy\_val = entropy(hist)

*# Predict the class label for the image*  img\_features = [[var, skewness, kurt, entropy\_val]] predicted\_class = model.predict(img\_features)

*# Print the predicted class label*

print("Predicted Class Label:", predicted\_class)

print("Predicted Class Probabilities 0 for fake|1 for original") print("var:", var) print("ske:", skewness) print("kurt Class Label:", kurt) print("entropy\_val:", entropy\_val)

*# output the prediction*  **if** predicted\_class <= 0.75:

print("Fake note") **else**: print("Real note")

detect\_image(predicted\_class,path)

*##############################*

**return** predicted\_class

*# Load the image* path = "Example.jpg" image = cv2.imread(path) *# Preprocess the image dand get reulst* feature\_vector = preprocess\_image(image)

# Text Description automatically generated with medium confidenceהדגמה

Graphical user interface

Description automatically generated with medium confidence

Text

Description automatically generated with low confidence

# הערכה

# Data Set

בתחילת הדרך היה ברשותנו DATASET רחב שהיה זמין לשימוש באתר Roboflow שהכיל מעל 1000 שטרות מקוריים וכ-200 שטרות מזויפים. מאגר נתונים זה היה מחולק באופן הבא:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 1$ | 2$ | 5$ | 10$ | 50$ | 100$ |
| מקורי | 200 | 200 | 200 | 200 | 200 | 200 |
| מזויף | 40 | 40 | 40 | 40 | 40 | 40 |

עבור כל תמונה במאגר זה, היה לנו גם קובץ label שהכיל את קואורדינטות של השטר בתמונה (ROI).

במהלך העבודה גילינו כי מאגר הנתונים של השטרות המזויפים הוסר מהאתר ונאלצנו לחפש חלופות חדשות.

לאחר התייעצות עם מנחת הפרויקט, הומלץ לנו לשנות את הגישה לבחינת אמינות השטרות בהתאם למגבלות מאגרי הנתונים הקיימים באינטרנט ולכן החלטנו לעבור לצורת בדיקה שונה עליה נפרט בהמשך.

מאגר הנתונים החדש שלנו אותו מצאנו באתר Kaggle [9] מכיל מידע על 1372 שטרות כאשר החלוקה היא 762 שטרות מזויפים ו-610 שטרות מקוריים.

סוג המידע הקיים במאגר זה הוא מידע סטטיסטי שנלקח משטרות מזויפים ומקוריים בהתאם על ידי שימוש במצלמה תעשייתית, התמונות שהופקו באמצעות מצלמה זו הן בגודל של 400400x פיקסלים, לאחר קבלת התמונה בוצע שינוי בקנה מידה אפור ברזולוציה של 660dpi.

# מדדים

עבור כל שטר שנמצא במאגר שלנו קיימים 5 נתונים סטטיסטיים אודות השטר:

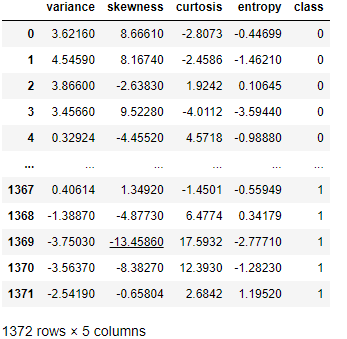
Variance – התפלגות הצבעים או רמות הבהירות על פני חלקים שונים של השטר

*Skewness* – צידוד המתייחס לאופן שבו הצבעים או רמת הבהירות מתפזרים על גבי השטר. אם רוב הצבעים או הבהירות "כהים" יותר אז המספרים יהיו חיובים, במידה והצבעים או הבהירות "בהירים" יותר אז המספרים יהיו שליליים.

*Kurtosis* – מונח סטטיסטי המתאר את אופן ההתפלגות של הנתונים, זוהי דרך למדוד את "כובד" הקצוות בהתפלגות (ראש וזנב).

*Entropy* – אנטרופיה בהקשר של תמונות מתייחסת לכמות האקראיות או "חוסר סדר" בהתפלגות הצבעים או ברמת הבהירות על פני חלקים שונים של השטר

*Class* – שטר מזויף 0, שטר מקורי 1.



Chart, histogram

Description automatically generated

Chart, scatter chart

Description automatically generated

Text, letter

Description automatically generated

Table

Description automatically generated

# צורת הבדיקה

בשלב ה- TEST, שבודק 20% (274 תמונות) הגענו ל85% דיוק, עם 10 חזרות (epochs), כאשר בהתחלה ניסינו מספר גבוה יותר אך קיבלנו התאמת-יתר (Overfitting).

# תוצאות

בבדיקה שערכנו, גילינו כי האלגוריתם מזהה על פי התכונות שפורטו בסעיף המדדים, ולכן ההבחנה מתבססת על גווני השטר ופיזורם. על סמך בדיקה של כ-100 שטרות מתוך מאגר תמונות אקראי של שטרות (80 אמיתיים ו-20 מזויפים), הגענו לרמת דיוק של 79 אחוזי דיוק.

# סיכום ומסקנות

התוצאות שהתקבלו מעט נמוכות משהצבנו כמטרה לפרויקט, אך אנו מבינים כי התוצאות נובעות מהאילוצים של מאגר הנתונים עליו התבססנו כפי שציינו בסעיף המתייחס לDATASET. טכניקת זיהוי התמונה מבוססת על המדדים המצוינים בסעיף המדדים, מדדים אשר מתבססים ברובם על גוונים ולא על תכונות כמו חספוס ואזורי עניין, כמצופה מבדיקת שטרות מזויפים ואמיתיים, אשר אינם נמצאים במאגר הנתונים. בנוסף, ניתן היה לראות כי האלגוריתם זיהה שטרות מקוריים בעיקר לפי צבעם ולכן שטר של 10 דולרים, שבטבעו בעל גוונים אדמדמים, זוהה כשטר מזויף במרבית הפעמים, אך יחד עם נתון זה הגענו לאחוזי הדיוק שצוינו בסעיף הקודם.

אנו סבורים שעם מאגר נתונים רחב יותר ובעל פרטים רלוונטיים יותר לבדיקת שטרות מזויפים ואמיתיים, ניתן להגיע לאחוזי דיוק גבוהים יותר, מאגר אשר כיום לא קיים.

עוד ידוע כי ישנה המגבלה שצוינה בתחילת הדו"ח, והיא המגבלה הטכנולוגית של טלפונים ניידים, בהם לא ניתן לבצע בדיקות של פס מגנטי, בדיקת עובי ואולטרה סגול, ולכן מראש אחוזי הדיוק אינן יכולים להיות קרובים למושלמים.

בשלב מגבלה זו, ניסינו למטב ככל שניתן את המודל עצמו. המיטוב בא לידי ביטוי בבדיקות מקדימות על המאגר, שיפור המודל באמצעות אלגוריתמים ושיטות סטטיסטיות לשיפור אחוזי הדיוק.

ברמה האישית, אנו יכולים להגיד שלמדנו המון על עולם למידת המכונה הן בפן התיאורטי והן בפן התכנותי, ואנו מאמינים שהצלחנו להעביר את הידע בדו"ח זה ובקוד עצמו.

# רשימת מקורות

[1] [Android-Based Verification System for Banknotes](https://www.proquest.com/docview/2124672974/DC10FB165E9B4E78PQ/13?accountid=143280)

[2] [Currency Detection for Visually Impaired Iraqi Banknote as a Study Case](https://www.proquest.com/docview/2623922087/DC10FB165E9B4E78PQ/4?accountid=143280)

[3] [Deep Learning-Based Detection of Fake Multinational Banknotes in a Cross-Dataset Environment Utilizing Smartphone Cameras for Assisting Visually Impaired Individuals](https://www.proquest.com/docview/2663027312/3BA2D4A846AA46A8PQ/2?accountid=143280&parentSessionId=0HkEMDUqOPZ%2FLpCuIxSOJ3VzpOIOHzK5EadmCQQS9jw%3D)

[4] [Banknote validation method for ATMs (automatic teller machines)](https://patents.google.com/patent/CN102915591A/en)

[5] [Joffermar - Gourme](http://www.mashkat.co.il/product-page/joffemar-gourme)

[6] [JetCount Money Counters](https://store.cumminsallison.com/products-supplies/jetcount-money-counters.html)

[7] [Cash Reader: Bill Ifentifier](https://play.google.com/store/apps/details?id=com.martindoudera.cashreader&hl=en&gl=US)

[8] [Banknote Scanner - AutuomaticB](https://play.google.com/store/apps/details?id=com.authority.banknotechecker&hl=en&gl=US&pli=1)

[9] [Bank Note Authentication UCI data](https://www.kaggle.com/datasets/ritesaluja/bank-note-authentication-uci-data)