1. Introduction

1.1 What is Machine Learning?

1.1.1 The Basic Concept

Artificial Intelligence (AI)

בינה מלאכותית הינו תחום בו תוכנות מחשב או מנגנון טכנולוגי אחר מחקה מנגנון חשיבה אנושי. בתחום רחב זה יש רמות שונות של בינה מלאכותית – יש מערכות שמסוגלות ללמוד דפוסי התנהגות ולהתאים את עצמן לשינויים, ואילו יש מערכות שאמנם מחקות מנגנון חשיבה אנושי אך הן לא מתוחכמות מעבר למה שתכנתו אותן בהתחלה. שואב רובוטי היודע לחשב את גודל החדר ואת מסלול הניקוי האופטימלי פועל לפי פרוצדורה ידועה מראש, ואין בו תחכום מעבר לתכנות הראשוני שלו. לעומת זאת תוכנה היודעת לסנן רעשים באופן מסתגל, או להמליץ על שירים בנגן מוזיקה בהתאם לסגנון של המשתמש, משתמשות בבינה מלאכותית ברמה גבוהה יותר, כיוון שהן לומדות עם הזמן דברים חדשים.

המונח בינה מלאכותית מתייחס בדרך כלל למערכת שמחקה התנהגות אנושית, אך היא שגרתית, לא לומדת משהו חדש, ועושה את אותו הדבר כל הזמן. מערכת זו יכולה להיות משוכללת ולחשב דברים מסובכים, אך תמיד עבור אותו הקלט (Input), יהיה אותו הפלט (Output).

ניקח לדוגמא מערכת סטרימינג של סרטים, למשל Netflix. כחלק משיפור המערכת והגדלת זמני הצפייה, ניתן לבנות מנגנון המלצות הבנוי על היסטוריית השימוש של הלקוחות שלי במערכת – איזה סרטים הם רואים, איזה ז'אנרים ומתי. כשיש מעט צופים ומעט סרטים, ניתן לעשות זאת באופן ידני – למלא טבלאות של הנתונים, לנתח אותם ידנית ולבנות מערכת חוקים שמהווה מנוע המלצות מבוסס AI. ניקח לדוגמא אדם שצופה ב"פארק היורה" וב"אינדיאנה ג'ונס" – סביר שהמערכת תמליץ לו לצפות גם ב-"פולטרגייסט". אדם שצופה לעומת זאת ב-"אהבה בין הכרמים" ו"הבית על האגם", ככל הנראה כדאי להמליץ לו על "הגשרים של מחוז מדיסון".

מערכת זו יכולה לעבוד טוב, אך בנקודה מסוימת כבר לא ניתן לנסח אותה כפרוצדורה מסודרת וכאוסף של חוקים ידוע מראש. מאגר הסרטים גדל, נוספים סוגים נוספים של סרטים (כמו למשל סדרות, תוכניות ריאליטי ועוד) ובנוסף רוצים להתייחס לפרמטרים נוספים – האם הצופה ראה את כל הסרט או הפסיק באמצע, מה גיל הצופה ועוד. מערכת הבנויה באופן קלאסי אינה מסוגלת להתמודד עם כמויות המידע הקיימות, וכמות הכללים שנדרש לחשוב עליהם מראש היא עצומה ומורכבת לחישוב.

נתבונן על דוגמא נוספת – מערכת לניווט רכב. ניתן להגדיר כלל פשוט בו אם משתמש יוצא מתל אביב ורוצה להגיע לפתח תקווה, אז האפליקציה תיקח אותו דרך מסלול ספציפי שנבחר מראש. מסלול זה לא מתחשב בפרמטרים קריטיים כמו מה השעה, האם יש פקקים או חסימות ועוד. כמות הפרמטרים שיש להתייחס אליהם איננה ניתנת לטיפול על ידי מערכת כללים ידועה מראש, וגם הפונקציונאליות המתאפשרת היא מוגבלת מאוד – למשל לא ניתן לחזות מה תהיה שעת ההגעה וכדומה.

Machine Learning (ML)

למידת מכונה הוא תת תחום של בינה מלאכותית, הבא להתמודד עם שני האתגרים שתוארו קודם – היכולת לתכנת מערכת על בסיס מסות של נתונים ופרמטרים, וחיזוי דברים חדשים כתלות בפרמטרים רבים שיכולים להשתנות עם הזמן. מנגנוני ML מנתחים כמויות אדירות של דאטה ומנסות להגיע לאיזו תוצאה. אם מדובר באפליקציית ניווט, המערכת תנתח את כל אותם הפקטורים ותנסה לחשב את משך הנסיעה המשוער. נניח והיא חזתה 20 דקות נסיעה. אם בסופו של דבר הנסיעה ארכה 30 דקות, האלגוריתם ינסה להבין איזה פקטור השתנה במהלך הדרך ומדוע הוא נכשל בחיזוי (למשל: הכביש בן ארבעה נתיבים, אבל במקטע מסיים הוא מצטמצם לאחד וזה מייצר עיכוב, וזה עיכוב קבוע ברוב שעות היממה ולא פקק אקראי). בהינתן מספיק מקרים כאלה, האלגוריתם "מבין" שהוא טועה, והוא פשוט יתקן את עצמו ויכניס למערך החישובים גם פקטור של מספר נתיבים ויוריד אולי את המשקל של הטמפרטורה בחוץ. וככה באופן חזרתי האלגוריתם שוב ושוב מקבל קלט, מוציא פלט ובודק את התוצאה הסופית. לאחר מכן הוא בודק היכן הוא טעה, משנה את עצמו, מתקן את המשקל שהוא נותן לפקטורים שונים ומשתכלל מנסיעה לנסיעה.

במערכות אלה ה-Input נשאר לכאורה קבוע, אבל ה-Output משתנה – עבור זמני יציאה שונים, האלגוריתם יעריך זמני נסיעה שונים, כתלות במגוון הפרמטרים הרלוונטיים.

מערכות ML משמשות את כל רשתות הפרסום הגדולות. כל אחת מנסה בדרכה שלה לחזות למשל, איזה משתמש שהקליק על המודעה צפוי שיבצע רכישה. הפלטפורמות השונות מנסות לזהות כוונה (Intent) על ידי למידה

מניסיון. בהתחלה הן פשוט ניחשו על פי כמה פקטורים שהוזנו להם על ידי בני אדם. נניח, גוגל החליטה שמי שצופה בסרטוני יוטיוב של Unboxing הוא ב-Intent גבוה של רכישה. בהמשך הדרך, בהנחה והמשתמש מבצע רכישה כלשהי, האלגוריתם מקבל "נקודה טובה". אם הוא לא קנה, האלגוריתם מקבל "נקודה רעה". ככל שהוא מקבל יותר נקודות טובות ורעות, האלגוריתם יודע לשפר את עצמו, לתת משקל גדול יותר לפרמטרים טובים ולהזניח פרמטרים פחות משמעותיים. אבל רגע, מי אמר למערכת להסתכל בכלל בסרטוני Unboxing?

האמת שהיא שאף אחד. מישהו, בנאדם, אמר למערכת לזהות את כל הסרטונים שמשתמש צופה בהם ביוטיוב, לזהות מתוך הסרטון, האודיו, תיאור הסרטון ומילות המפתח וכו' – איזה סוג סרטון זה. ייתכן ואחרי מיליארדי צפיות בסרטונים, האלגוריתם מתחיל למצוא קשר בין סוג מסוים של סרטונים לבין פעולות כמו רכישה באתר. באופן הזה, גוגל מזינה את האלגוריתם בכל הפעולות שהמשתמש מבצע. המיילים שהוא קורא, המקומות שהוא מסתובב בהם, התמונות שהוא מעלה לענן, ההודעות שהוא שולח, כל מידע שיש אליו גישה. הכל נשפך לתוך מאגר הנתונים העצום בו מנסה גוגל לבנות פרופילים ולמצוא קשר בין פרופיל האדם לבין הסיכוי שלו לרכוש או כל פעולה אחרת שבא לה לזהות.

המכונה המופלאה הזו לומדת כל הזמן דברים חדשים ומנסה כל הזמן למצוא הקשרים, לחזות תוצאה, לבדוק אם היא הצליחה, ואם לא לתקן את עצמה שוב ושוב עד שהיא פוגעת במטרה. חשוב לציין שלמכונה אין סנטימנטים, כל המידע קביל ואם היא תמצא קשר מוכח בין מידת הנעליים של בנאדם לבין סרטונים של בייבי שארק, אז היא תשתמש בו גם אם זה לא נשמע הגיוני.

חשוב לשים לב לעניין המטרה – המטרה היא לא המצאה של האלגוריתם. הוא לא קם בבוקר ומחליט מה האפליקציה שלכם צריכה לעשות. המטרה מוגדרת על ידי היוצר של המערכת. למשל – חישוב זמן נסיעה, בניית מסלול אופטימלי בין A ל-B וכו'. המטרה של גוגל – שמשתמש יבצע רכישה, והכל מתנקז לזה בסוף, כי גוגל בראש ובראשונה היא מערכת פרסום. אגב, גם ההגדרה של מסלול "אופטימלי" היא מעשה ידי אדם. המכונה לא יודעת מה זה אופטימלי, זו רק מילה. אז צריך לעזור לה ולהגיד לה שאופטימלי זה מינימום זמן, מעט עצירות, כמה שפחות רמזורים וכו'. לסיכום, המטרה מאופיינת על ידי האדם ולא על ידי המכונה. המכונה רק חותרת למטרה שהוגדר לה.

יש מנגנוני ML המתבססים על דאטה מסודר ומתויג כמו ב-Netflix, עם כל המאפיינים של הסרטים אבל גם עם המאפיינים של הצופים (מדינה, גיל, שעת צפייה וכו'..). לעומת זאת יש מנגנוני ML שמקבלים טיפה יותר חופש ומתבססים על מידע חלקי מאד (יש להם מידע על כל על הסרטים, אבל אין להם מידע על הצופה). מנגנונים אלו לא בהכרח מנסים לבנות מנוע המלצות אלא מנסים למצוא חוקיות בנתונים, חריגות וכו'.

כך או כך, המערך הסבוך הזה הקרוי ML בנוי מאלגוריתמים שונים המיומנים בניתוח טקסט, אלגוריתמים אחרים המתמקדים בעיבוד אודיו, כאלה המנתחים היסטוריית גלישה או זיהוי מתוך דף ה-Web בו אתם צופים ועוד. עשרות או מאות מנגנונים כאלה מסתובבים ורצים ובונים את המפה השלמה. ככה רוב רשתות הפרסום הגדולות עובדות. ככל שהמכונה של גוגל/פייסבוק תהיה חכמה יותר, ככה היא תדע להציג את המודעה המתאימה למשתמש הנכון, בזמן הנכון ועל ה-Device המתאים.

1.1.2 Data, Tasks and Learning

כאמור, המטרה הבסיסית של למידת מכונה היא היכולת להכליל מתוך הניסיון, ולבצע משימות באופן מדויק ככל הניתן על דאטה חדש שעדיין לא נצפה, על בסיס צבירת ניסיון מדאטה קיים. באופן כללי ניתן לדבר על שלושה סוגים של למידה:

למידה מונחית – הדאטה הקיים הינו אוסף של דוגמאות, ולכל דוגמא יש תווית (label). מטרת האלגוריתמים במקרה זה היא לסווג דוגמאות חדשות שלא נצפו בתהליך הלמידה. באופן פורמלי, עבור דאטה $x \in \mathbb{R}^{n\times d}$, יש סט labels – $y \in \mathbb{R}^{1\times d}$, ומחפשים את האלגוריתם שמבצע את המיפוי $g: X \to Y$ בצורה הטובה ביותר, כלומר בהינתן דוגמא חדשה $x \in \mathbb{R}^n$, המטרה היא למצוא עבורה את ה-y הנכון. המיפוי נמדד ביחס לפונקציות מחיר, כפי שיוסבר בהמשך בנוגע לתהליך הלמידה.

למידה לא מונחית – הדאטה הקיים הינו אוסף של דוגמאות במרחב, בלי שנתון עליהן מידע כלשהו המבחין ביניהן. במקרה זה, בדרך כלל האלגוריתמים יחפשו מודל המסביר את התפלגות הנקודות – למשל חלוקה לקבוצות שונות וכדומה.

למידה באמצעות חיזוקים – הדאטה בו נעזרים אינו מצוי בתחילת התוכנית אלא נאסף עם הזמן. ישנם סוכנים הנמצאים בסביבה מסוימת ומעבירים מידע למשתמש, והוא בתורו ילמד אסטרטגיה בה הסוכנים ינקטו בצעדים הטובים עבורם.

האלגוריתמים השונים של הלמידה מתחלקים לשתי קבוצות – מודלים דיסקרימנטיביים המוציאים פלט על בסיס מידע נתון, אך לא יכולים ליצור מידע חדש בעצמם, ומודלים גנרטיביים, שלא רק לומדים להכליל את הדאטה הנלמד גם עבור דוגמאות חדשות, אלא יכולים גם להבין את מה שהם ראו וליצור מידע חדש על בסיס הדוגמאות שנלמדו.

כאמור, בשביל לבנות מודל יש צורך בדאטה. מודל טוב הוא מודל שמצליח להכליל מהדאטה הקיים גם לדאטה חדש. המודל למעשה מנסה למצוא דפוסים בדאטה הקיים, מהם הוא יוכל להסיק מסקנות גם על דוגמאות חדשות. כדי לוודא שהמודל אכן מצליח להכליל גם על דוגמאות חדשות, בדרך כלל מחלקים את הדאטה הקיים לשניים – סט אימון (training set) וסט מבחן (test set). אם המודל מצליח למצוא דפוסים בסט האימון שנכונים גם עבור הטסט סט, זה סימן שהמודל הצליח למצוא כללים שיכולים להיות נכונים גם לדוגמאות חדשות שיבואו.

1.2 Applied Math

האלגוריתמים של למידת מכונה נסמכים בעיקרם על שלושה ענפים מתמטיים; אלגברה לינארית, חשבון דיפרנציאלי והסתברות. בפרק זה נציג את העקרונות הנדרשים בלבד, ללא הרחבה, על מנת להבין את הנושאים הנדונים בספר זה.

1.2.1 Linear Algebra

<u>וקטורים ומרחבים וקטורים</u>

באופן מתמטי מופשט, וקטורים, המסומנים בדרך כלל ע"י $\vec{x}$ או על ידי $\boldsymbol{x}$, הינם אובייקטים הנמצאים במרחב וקטורי $(V,+,\cdot)$ מעל שדה $\mathbb{F}$. מהו אותו מרחב וקטורי?

ראשית השדה, $\mathbb{F}$, הוא קבוצת מספרים המקיימים תכונות מתמטיות מסוימות. לדיון בספר זה, השדה הוא קבוצת המספרים הממשיים, שלעיתים גם מסומנת באות $\mathbb{R}$. שנית, נשים לב כי המרחב הוקטורי דורש גם הגדרת פעולת חיבור $(+)$.

כעת, $(V,+)$ היא מרחב וקטורי אם הוא מקיים את התכונות הבאות:

(I) קיים איבר אפס (וקטור אפס) כך שכל $\vec{x}$ בקבוצה V מקיים: $\vec{x}+\vec{0}=\vec{0}+\vec{x}=\vec{x}$.
(II) לכל איבר בשדה a ולכל $\vec{x}$ ו-$\vec{y}$ בקבוצה V, גם $a\cdot\vec{x}+\vec{y}$ הינו גם איבר בקבוצה V.

הערה: קיימות דרישות נוספות למרחב וקטורי, אך הם מעבר לנדרש בספר זה.

דוגמאות:

א. וקטורים גאומטריים:
מערך חד מימדי $\vec{x}=(x_1,x_2\dots x_n)$ (n-יה סדורה) נקרא וקטור גאומטרי n מימדי, כאשר רכיבי הוקטור הם איברים בשדה $\mathbb{F}$. האיבר x_i, המיוצג על ידי האינדקס i מתאר את מיקום האיבר. מרחב זה מסומן ע"י $\mathbb{F}^n$.
נוכיח שמרחב זה הוא אכן מרחב וקטורי:
<u>חיבור וקטורים:</u>

$$\vec{x}=(x_1,x_2\dots x_n),\quad \vec{y}=(y_1,y_2,\dots,y_n)\quad\rightarrow\quad \vec{x}+\vec{y}=(x_1+y_1,x_2+y_2,\dots,x_n+y_n)$$

<u>וקטור אפס:</u>

$$\vec{0}=(0,0,\dots,0)$$

<u>כפל בסקלר:</u>

$$\vec{x}=(x_1,x_2\dots x_n)\quad\rightarrow\quad a\,\vec{x}=(a\,x_1,a\,x_2,\dots,a\,x_n)$$

הערה: לשם פשטות, בהמשך, נכנה וקטור גאומטרי כ"וקטור" בלבד.

ב. מטריצות:
מערך דו מימדי $\begin{pmatrix} A_{11} & \dots & A_{1m} \\ \vdots & \ddots & \vdots \\ A_{n1} & \dots & A_{nm} \end{pmatrix}$, אשר רכיביו הם איברים בשדה $\mathbb{F}$, נקרא מטריצה מסדר $n\times m$, כאשר n הוא מספר השורות ו-m הוא מספר העמודות במערך. האיברים במטריצה A_{ij} מיוצגים ע"י שני אינדקסים – i,j, המתארים את השורה והעמודה בהתאמה. מרחב זה מסומן בדרך כלל ע"י $\mathbb{F}^{n\times m}$.
נוכיח שמרחב זה הוא אכן מרחב וקטורי:
<u>חיבור מטריצות:</u>

$$\hat{A}=\begin{pmatrix} A_{11} & \dots & A_{1m} \\ \vdots & \ddots & \vdots \\ A_{n1} & \dots & A_{nm} \end{pmatrix}, \hat{B}=\begin{pmatrix} B_{11} & \dots & B_{1m} \\ \vdots & \ddots & \vdots \\ B_{n1} & \dots & B_{nm} \end{pmatrix} \rightarrow \hat{A}+\hat{B}=\begin{pmatrix} A_{11}+B_{11} & \dots & A_{1n}+B_{1n} \\ \vdots & \ddots & \vdots \\ A_{m1}+B_{m1} & \dots & A_{nm}+B_{nm} \end{pmatrix}$$

<u>מטריצת אפס:</u>

$$\hat{0}=\begin{pmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{pmatrix}$$

<u>כפל בסקלר:</u>

$$\hat{A}=\begin{pmatrix} A_{11} & \dots & A_{1m} \\ \vdots & \ddots & \vdots \\ A_{n1} & \dots & A_{nm} \end{pmatrix} \quad \rightarrow \quad a\,\hat{A}=\begin{pmatrix} a\,A_{11} & \dots & a\,A_{1m} \\ \vdots & \ddots & \vdots \\ a\,A_{n1} & \dots & a\,A_{nm} \end{pmatrix}$$

ניתן להבחין כי הווקטורים הגיאומטריים שהוגדרו בדוגמא א, הם בעצם מטריצות במימד $n=n\times 1$.

ג. פולינומים:
פולינומים מסדר n הינם ביטויים מהסוג $a_0+a_1x+a_2x^2+\cdots a_nx^n$, כאשר n מייצג את החזקה הגדולה ביותר ו-a_i הם איברים בשדה. מרחב זה מסומן בדרך כלל ע"י $P_n(x)$.

בכל הדוגמאות לעיל קל להראות שהן אכן מהוות מרחב וקטורי. רשימה חלקית לדוגמאות נוספות לווקטורים (ולמרחבים וקטורים) כוללת למשל מרחבי פונקציות או אפילו אותות אלקטרומגנטיים. כאן בחרנו להציג רק את הדוגמאות הרלוונטיות לספר זה.

<u>פעולות חשבון על מטריצות ווקטורים:</u>

כמו שנזכר לעיל, הווקטורים הגיאומטריים שהוגדרו בדוגמא א, הם בעצם מטריצות במימד $n=n\times 1$. לכן, פעולות החשבון מוגדרות באופן זהה.

- <u>חיבור וחיסור בין שני מטריצות:</u>

$\hat{A},\hat{B}\in\mathbb{F}^{n\times m}$ כאשר A_{ij},B_{ij} הם האיברים בשורה i בעמודה j של המטריצות $\hat{A},\hat{B}$ בהתאמה. אז, האיבר בשורה i בעמודה j של מטריצת הסכום (או ההפרש) הינו

$$(A\pm B)_{ij}=A_{ij}\pm B_{ij}$$

(הגדרת חיבור המטריצות בעצם כבר ניתנה בדוגמא א לעיל)

שים לב: ניתן לחסר ולחסר מטריצות רק בעלות אותו המימד.

- <u>כפל בין שתי מטריצות:</u>

$\hat{A}\in\mathbb{F}^{n\times k},\hat{B}\in\mathbb{F}^{k\times m}$ הן שתי מטריצות, כאשר מספר השורות במטריצה $\hat{A}$ שווה למספר העמודות של מטריצה $\hat{B}$ (אך שתי המטריצות אינן בהכרח בעלות אותו מימד). במקרה כזה, מכפלת המטריצות מוגדרת על ידי:

$$\hat{A}\cdot\hat{B}=\begin{pmatrix} A_{11}B_{11}+A_{12}B_{21}+\cdots A_{1k}B_{k1} & \dots & A_{11}B_{1n}+A_{1k}B_{kn} \\ \vdots & \ddots & \vdots \\ A_{m1}B_{11}+\cdots+A_{mk}B_{kn} & \dots & A_{n1}B_{1m}+\cdots A_{nk}B_{km} \end{pmatrix}$$

למעשה כל איבר בתוצאה הינו סכום של מכפלת שורה i ממטריצה A בעמודה j ממטריצה B:

$$\left(\hat{A}\cdot\hat{B}\right)_{ij}=\sum_k A_{ik}B_{kj}$$

שים לב: על מנת שכפל המטריצות יהיה מוגדר מספר העמודות ב-$\hat{A}$ שווה למספר השורות ב$\hat{B}$.

עבור מטריצות ריבועיות (מסדר $n\times n$), מוגדר גם הכפל $\hat{A}\hat{B}$ וגם הכפל $\hat{B}\hat{A}$, אולם ייתכן ו $\hat{A}\hat{B}\neq\hat{B}\hat{A}$.

- <u>שחלוף:</u>

החלפת שורות בעמודות, או 'סיבוב' המטריצה. נניח מטריצה $\hat{A}\in\mathbb{F}^{n\times m}$, אז השחלוף שלה, המסומן כ-$\hat{A}^T$ הוא:

$$\left(\hat{A}^T\right)_{ij} = A_{ji}$$

ובאופן מפורש:

$$\hat{A} = \begin{pmatrix} A_{11} & \dots & A_{1m} \\ \vdots & \ddots & \vdots \\ A_{n1} & \dots & A_{nm} \end{pmatrix} \rightarrow \hat{A}^T = \begin{pmatrix} A_{11} & \dots & A_{n1} \\ \vdots & \ddots & \vdots \\ A_{1m} & \dots & A_{nm} \end{pmatrix}$$

שים לב שהמטריצה החדשה; $\hat{A}^T$ הינה במימד $m \times n$. בנוסף ניתן להוכיח כי מתקיים: $\hat{A} = \left(\hat{A}^T\right)^T$.

שחלוף של וקטור שורה, נותן וקטור עמודה ולהפך.

- <u>מטריצת יחידה:</u>

מטריצת יחידה, הינה מטריצה ריבועית (מסדר $n \times n$), המסומנת על ידי $\mathbb{I}_n$ ומוגדרת:

$$(\mathbb{I}_n)_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

ובאופן מפורש:

$$\mathbb{I}_n = \begin{pmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{pmatrix}$$

מטריצה זו מקיימת $\hat{A} \cdot \mathbb{I}_m = \mathbb{I}_n \cdot \hat{A} = \hat{A}$ לכל מטריצה $\hat{A}$ מסדר $n \times m$.

הערה: לעיתים סדר מטריצת היחידה אינו משנה או טריוויאלי, ולכן המטריצה מסומנת רק על ידי $\mathbb{I}$ ללא ציון המימד.

- <u>מטריצה הופכית:</u>

למטריצות ריבועיות (מטריצות עם מספר זהה של שורות ועמודות; מסדר $n \times n$) ייתכן ויש מטריצה הופכית $\hat{A}^{-1}$ שמקיימת את הקשר:

$$\hat{A} \cdot \hat{A}^{-1} = \hat{A}^{-1} \cdot \hat{A} = \mathbb{I}_n$$

דוגמא: $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \mathbb{I}_2$. לכן , במקרה הזה $\hat{A} = \hat{A}^{-1} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$.

<u>מערכת משוואות לינאריות:</u>

מערכת משוואות לינאריות מוצגת באופן כללי באופן הבא:

$$\begin{matrix} A_{11}x_1 + A_{12}x_2 + & \dots & A_{1n}x_n = b_1 \\ \vdots & \ddots & \vdots \quad \vdots \\ A_{m1}x_1 + A_{m2}x_{2+} & \cdots & A_{mn}x_n = b_m \end{matrix}$$

נשים לב כי מערכת משוואות לינארית ניתנת לייצוג באופן קומפקטי על ידי הפרדה בין רשימת המשתנים, המקדמים של משתנה, והאיבר החופשי, באופן הבא:

$$\hat{A}\,\vec{x} = \vec{b} = \begin{pmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & \ddots & \vdots \\ A_{m1} & \cdots & A_{mn} \end{pmatrix}\begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$$

מטריצה $\hat{A} = \begin{pmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & \ddots & \vdots \\ A_{m1} & \cdots & A_{mn} \end{pmatrix}$ הינה מטריצת המקדמים מסדר $n \times m$, כאשר n הוא מספר המשתנים, ו-m הוא מספר המשוואות במערכת.

וקטור $\vec{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$, הינו וקטור עמודה (לעיתים גם מסומן על ידי $(x_1, x_2 \dots x_n)^T$), המייצג את וקטור המשתנים.

וקטור $\vec{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}$, הינו וקטור עמודה, שאיבריו הם האיבר החופשי.

הפתרונות של מערכת המשוואות הליניארית, $\hat{A}\,\vec{x} = \vec{b}$, באם הם קיימים ויחידים, נתונים ע"י $\vec{x} = \hat{A}^{-1}\vec{b}$.

אורתוגונליות

אורתוגונליות היא הכללה של תכונת הניצבות המוכרת מגאומטריה. בגאומטריה, שני ישרים במישור האוקלידי ניצבים זה לזה אם הזווית הנוצרת בנקודת החיתוך שלהם היא זווית ישרה (בת 90 מעלות). מושג האורתוגונליות מכליל כונה זו גם למרחבים ווקטורים n-מימדים. על מנת להכליל את מושג הניצבות יש ראשית להגדיר זווית בין שני וקטורים:

$$\cos(\theta) = \frac{\langle x, y \rangle}{\|x\| \cdot \|y\|}$$

כאשר $\langle x, y \rangle$ הינה המכפלה הפנימית בין שני הווקטורים, המוגדרת מעל הטבעיים כך: $\langle x, y \rangle = \sum_i x_i \cdot y_i$, והביטוי במכנה $\|x\| \cdot \|y\|$ הוא מכפלת הנורמות. לפי אי שיוויון קושי שוורץ ניתן להוכיח שהביטוי באגף ימין תמיד קטן או שווה בערכו המוחלט ל-1, ולכן לפי ההגדרה הזו תמיד ניתן לחשב זווית בין שני וקטורים.

לווקטורים אורתוגונליים חשיבות רבה כאשר חוקרים מרחבים וקטורים. לבסיס של מרחב וקטורי יש מספר תכונות נוחות כאשר הוא אורתונורמלי (כל אבריו אורתוגונליים זה לזה ובעלי אורך 1). יתר על כן, מתברר שבהינתן בסיס כלשהו למרחב וקטורי ניתן לקבל ממנו בסיס חדש שכל אבריו אורתוגונליים זה לזה, כך שתמיד ניתן למצוא בסיס נוח שכזה. דבר זה נעשה על ידי תהליך גרם-שמידט.

שני וקטורים אורתוגונליים יסומנו על ידי $\perp$. עבור וקטורים אורתוגונליים מתקיימות התכונות הבאות:

- אם $u \perp v$, אז $v \perp u$.
- אם $u \perp v$, אז לכל סקלר λ גם $\lambda u \perp v$.
- אם $u \perp v$ וגם $w \perp v$, אז $(w + u) \perp v$.
- אם וקטור אורתוגונלי לקבוצה של וקטורים אזי הוא גם אורתוגונלי לכל צירוף לינארי שלהם (נובע משתי התכונות הקודמות).

וקטורים עצמיים וערכים עצמיים

תהי $A \in \mathbb{F}^{n \times n}$ מטריצה ריבועית, וקטור $v \in \mathbb{F}^n$ ו-$\lambda \in \mathbb{F}$ סקלר. λ נקרא ערך עצמי של A ו-v נקרא הערך העצמי המתאים אם מתקיים:

$$A \cdot v = \lambda \cdot v$$

ניתן להראות שעבור מטריצה A, הוקטורים העצמיים המתאימים לסקלר λ הם כל פתרונות המשוואה ההומוגנית $(A - \lambda \mathbb{I}_n)v = 0$.

אם נסמן $V = [v_1, \dots, v_n]$, ו-$\Lambda = [\lambda_1, \dots, \lambda_n]$, אזי מתקיים:

$$A = V \operatorname{diag}(\Lambda)\, V^{-1}$$

כאשר $\operatorname{diag}(\Lambda)$ הוא ערכי האלכסון של המטריצה Λ.