**רשת ניורונים וmachine learning**

מאת

**עלי חאג ולין סחניני**

**עבודה במדע חישובי י"ב**

**עבודה זו בוצעה בהדרכת: אריק בר חיים**

**מבוא:**

**מה זו בינה מלאכותית?**

האינטליגנציה המלאכותית או בינה מלאכותית (Artificial Intelligence) היא מכונה שיכולה לפתור את כל הבעיות שאדם יכול לפתור. בניגוד למחשבים של היום, שמסוגלים להתמודד עם בעיות חישוביות, אך לא לחשוב, מערכת כזו תוכל לבצע חשיבה של ממש, כולל חשיבה מופשטת (על מושגים), דמיון, הסקת מסקנות לוגיות, ראייה, הבנת שפה טבעית, דיבור, הבנת העולם והכי חשוב, למידה. מערכת כזו תוכל ללמוד ולהתפתח מבחינת יכולותיה, ממש כמו שאנו לומדים כל הזמן וכך נעשים חכמים יותר. כך תפעל המערכת על פי הבנתה.

בינה מלאכותית יכולה לחשוב. שימוש במידע שנאסף מבחוץ (בעזרת החושים) וזה שבפנים (ממאגרי הזיכרון והידע שצברנו בעבר) מאפשר את החשיבה. לחשוב פירושו להשתמש במידע הזה ולארגן אותו מחדש ובכך ליצור מידע חדש. המידע שנוצר הוא הלמידה שלנו, ההמצאות שלנו, היצירות האמנותיות, ההומור וכדומה.

**Machine Learning**

הוא תחום שאמנם קיים בערך משנות ה-60, אך שהתפתח בעיקר ב 10-15 השנים האחרונות.

מנקודת מבט אחת, Machine Learning הוא כלי שמאפשר לפתור בעיות תוכנה, שלא ניתן לפתור בדרך הרגילה.

לא כל בעיה ניתן לפתור בעזרת - Machine Learning להיפך. הבעיות העיקריות שניתן לפתור בעזרת ML מחולקות לקטגוריות הבאות:

* **Regression**-חיזוי של ערך, על בסיס נתונים. למשל: שערוך ערך של דירה על סמך הנתונים שלה (שטח, גיל, וכו’) או הערכה מה יהיה סכום העסקאות של לקוח בחודש הקרוב – על סמך עסקאות עבר, גיל, ומדדים של הרגלי קנייה.
* **Classification**- סיווג של entities למספר סגור של קבוצות מוגדרות.
* **Clustering**- גיבוש קבוצות של פריטים מסוימים.
* **Recommender Systems**  - חיזוי פריטים דומים שמישהו עשוי להתעניין בהם,
* **Collaborative Filtering -**  ( מי שאהב X אהב גם Y ) המלצת תוכן או מוצר למשתמש, ע”פ סיווג של משתמשים “דומים”, ותיאור הפריטים שקבוצת ה”משתמשים הדומים” נוטה לאהוב.

את עולם ה Machine Learning מחלקים בדרך ל-2 קטגוריות:

**1.**למידה מפוקחת (supervised learning)

כאשר יש לנו מאגר של נתונים ותשובות (target value), שהושגו בצורה כזו או אחרת (מדידה, עבודה של מומחים, וכו’) – ואנו רוצים שהמחשב ינסה לחקות את התשובות.

דוגמה קלאסית: שערוך ערך של דירה. גם אם יש לנו נתונים של אלפי דירות ומחיר מכירה של כל אחד מהן – זו תהיה משימה כמעט בלתי אפשרית לכתוב קוד בעזרת if ו-for שיבצע שיערוך של מחיר דירה, ובצורה טובה.

**2.**למידה לא מפוקחת (unsupervised learning)

זוהי קטגוריה מעט שונה של Machine Learning בה יש לנו סט של נתונים, אך ללא תשובות (target value) מהם אנו מנסים להפיק תובנות.

למשל: אנו מנסים לסווג מאמרים לקבוצות הגיוניות, מבלי שיש דוגמה לסיווג שכזה.

ניתן להשתמש ב- features כמו אורך המאמר, מספר המלים שחוזר פעמים רבות יותר בכל מאמר, או המצאות / אי המצאות של מילות מפתח.

גם כאן, אנו בוחרים אלגוריתם שיבנה לנו מודל, לו נזין נתונים של מאמרים – ונקבל הצעה לחלוקה לקבוצות.

ההערכה של הצלחת המודל תהיה תהליך קצת שונה: אולי תהליך ידני, אולי הרצה של סימולטור כזה או אחר שינסה לקבוע את הצלחת המודל.

* **תהליך העבודה (Workflow) של Machine Learning ל Supervised Learning כמו בעיית רגרסיה:**

**1) הגדרת הבעיה --> כיוון הפתרון**

**2) השגת הנתונים (Obtaining Data)**

יש להשיג נתונים “מוצלחים”, ולא סתם נתונים “באיזור הבעיה”. זה אומר הרבה פעמים להתחיל לאסוף נתונים שלא נאספו בעבר, ולהבין המשמעות המדויקת של כל נתון. לכמות הנתונים יש גם חשיבות גדולה: אלגוריתמים רבים יהיו מוצלחים הרבה יותר – כאשר יש להם יותר נתונים ללמוד מהם**.**

**3) קרצוף הנתונים (Scrubbing Data)**

גם ברגע שיש נתונים “נכונים “- לרוב יהיו להם בעיות איכות (Data Quality):

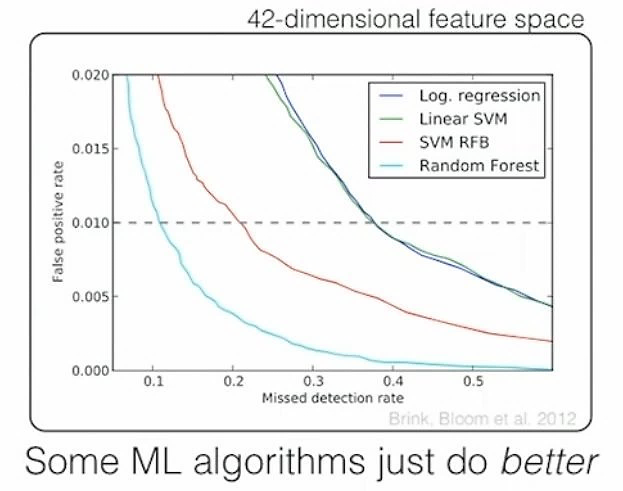
* להשלים ערכים חסרים (או להסיר אותם, אם תיקון לא אפשרי).
* לתקן אי התאמות.
* נורמליזציה של הנתונים.

שלב השגת הנתונים + “הקרצוף” נחשבים בד”כ לכ 50-80% מכלל העבודה ב ML Workflow – החלק הארי. שלב זה דורש יכולות תכנות, קצת הבנה סטטיסטית (עבור תיקונים וקירובים), ומעט Domain Knowledge – על מנת להבין את הנתונים טוב יותר. כמעט תמיד ניתן להשקיע עוד עבודה בכדי לשפר עוד יותר את איכות הנתונים – וחשוב לדעת גם מתי “לעצור”.

**4) בחירת האלגוריתם**

ישנם עשרות אלגוריתמים של Machine Learning, וכמה שרלוונטיים לכל סוג של בעיה. עלינו להבין את הנתונים ואת האלגוריתם – על מנת לנסות ולהתאים את האלגוריתם הטוב ביותר.

הבחירה באלגוריתם תשפיע על רמת הדיוק של התוצאה, על מהירות הריצה, על הרגישות לכמות ולאיכות הנתונים, על קלות הבנת התוצאות של האלגוריתם, ועל ההתאמה לבעיה הספציפית וסוג הנתונים שיש לנו, ועוד.



בתרשים למעלה ניתן לראות כיצד בחירה של אלגוריתם משפיעה על אחוז הטעות של האלגוריתם.

**5) אימון המודל**

זהו השלב בו מזרימים את הנתונים לאלגוריתם, ובד”כ בונים תהליך אוטומטי שיעשה זאת.

נרצה לנסות אימון של המערכת עם קבוצות שונות של features וכך לבדוק אילו features הם מוצלחים יותר לפתרון הבעיה. זהו סוג של fine tuning של האלגוריתם.

לרוב, פחות features יובילו לכמות קטנה יותר של טעויות ולעתים קרובות הנתונים שלנו יתעדכנו בצורה שוטפת, וברגע שנתגבש על מודל שעובד – נרצה לאמן מחדש את המערכת כל שבוע, או אפילו כל יום.

**Deep Learning**

**מה עושות רשתות נוירוניות?**

רשתות נוירונים הן רשתות מחשבים מתקדמות שמחקות את החשיבה האנושית. נוירון הוא תא עצב. רשת נוירונים היא "רשת עצבית" מלאכותית, שמעתיקה את הפעילות של שכבות תאי העצב בניאו-קורטקס, האזור שתופס את רוב המוח האנושי ושבו נעשית החשיבה שלנו. כל נוירון מחובר בסינפסות אל נוירונים נוספים, במה שקרוי "רשת נוירונית". התקשורת בין הנוירונים היא באמצעות "פוטנציאלי פעולה" - מעין הבזקים, פעימות חשמליות, שעוברות בסינפסות בין נוירון, תא עצב, למשנהו והלאה אל הנוירון הבא ולאלה שאחריו.

מחקר רשתות הנוירונים עוסק בבניית מודלים וירטואליים של שכבות תאי העצב ובתחום זה עוסקים כיום צוותים מגוונים, שמורכבים ממדעני מוח, מדעני מחשב ותוכנה. רשתות מחשבים כאלה יוכלו בעתיד לבצע פעולות שהמוח יודע לבצע בקלות: לאסוף מידע, לנתחו ולהגיב לו בקבלת החלטות שאנו עושים אלפי פעמים ביום.

תחום הנוירו-מחשוב הוא אחד התחומים החשובים בעולם כיום. הוא יוביל בעתיד לדור מתקדם במיוחד של תוכנות חכמות. דמיינו את האפשרויות של תוכנות-מוח שכאלה - מאפליקציות שיידעו את טעמנו האישי ויציעו לנו בגדים, אוכל, מוסיקה או ספרים שאנו אוהבים ועד טלפונים חכמים, שילמדו הרגלים, תחביבים, העדפות וצרכים של המשתמש ויסייעו לו. דמיינו מצלמות חכמות, שיודעות לזהות פורץ או מבקר תמים בביתנו, מערכות מחשוב שיזהו האקרים לעומת משתמשים תמימים שטעו, בידוק בטחוני שיזהה מפגים או עבריינים ומערכות צבאיות שיוודאו שכוחותינו יזהו וישמידו את כוחות האויב ולא את כוחותינו.

**רשת ניורונים**

נוירונים מלאכותיים מאורגנים בשכבות המחוברות על ידי קישורים המרכיבים רשת של נוירונים.

השכבה הראשונה - שכבת הinput - מספקת נתונים מדבר אחד הזקוק לסיווג, זה באמצעות הfeatures ובהזנת מספרים בתוך ה feature.

בקצה השני יש שכבת פלט( output ) שמספר הנוירונים שלה שווה למספר התוצאות האפשריות.

בין לבין יש לנו שכבה נסתרת אשר הופכת input לתוך .output

הדבר הראשון שנוירון עושה הוא להכפיל את כל קלט על ידי משקל מסוים. לאחר מכן הוא מסכם את הinput המשוקללות האלה יחד, ואז הניורון מיישם הטייה לתוצאה, כלומר הוא מוסיף או מחסר ערך קבוע.

ההטיות והקלט השקול נקבעים תחילה לערכים אקראיים כאשר נוצרת רשת עצבית, ואז אלגוריתם נכנס ומתחיל לשפץ את כל אותם ערכים כדי להכשיר את הרשת העצבית באמצעות נתונים שכותרתו לאימון ובדיקה. זה קורה על פני אינטראקציות רבות בהדרגה לשפר את הדיוק. תהליך הדומה ללמידה אנושית.

לבסוף יש לניורונים פונקציית הפעלה המכונה גם פונקציית העברה, שמוחלת על הפלט ומבצעת שינוי מתמטי סופי לתוצאה. לדוגמה, הגבלת הערך לטווח של מינוס אחד לאחד או הגדרת כל ערך שלילי לאפס.

השכבה המוסתרת לא חייבת להיות רק שכבה אחת, היא יכולה להיות הרבה שכבות עמוקות וזה המקום שבו המונח למידה עמוקה מגיע.

**רגרסיה לינרית (Linear Regression):**

**רגרסיה לינארית** היא שיטה מתמטית למציאת הפרמטרים של הקשר בין משתנה בלתי תלוי X למשתנה תלוי, Y בהנחה שהקשר ביניהם לינארי, כלומר מהצורה . Y=aX+b

השיטה משמשת לניתוח מדגמים סטטיסטיים. נוסחת הרגרסיה הלינארית מחשבת את הקו הישר שעובר דרך הנקודות שבמדגם. במצב של קשר ישיר מדויק כל הנקודות במדגם יימצאו על הקו עצמו.

במציאות גורמים נוספים משפיעים על המדגם והנקודות לרוב מפוזרות מסביב לקו. הקו מחושב בצורה כזאת שסכום ריבועי המרחקים של הנקודות מהקו הוא הקטן ביותר. רגרסיה לינארית מרובה מחשבת קשר בין מספר משתנים בלתי תלויים יחד, למשתנה תלוי אחד.

**: Gradient Decent**

GradientDescent משמש בעת אימון מודל הלמידה של המחשב. זהו אלגוריתם אופטימיזציה, המבוסס עלconvex function , שמשפר את הפרמטרים באמצעות איטרציה כדי למזער פונקציה נתונה למינימום המקומי שלה.

זה פשוט משמש כדי למצוא את הערכים של מקדמי הפונקציות כדי להקטין את ה cost function ככל האפשר.

אנחנו מתחילים על ידי הגדרת ערכי הפרמטרים הראשוניים ומשם ה gradient descent מתאים באמצעות אטרציה את הערכים, כך שהם יעשו מינימזציה לcost function .

שיפוע מודד כמה הפלט של פונקציה משתנה אם נשנה את הinput קצת.

אנחנו יכולים גם לחשוב על gradient כמו שיפוע של פונקציה. ככל שה gradient יותר גדול, המדרון תלול יותר והמודל יכול ללמוד יותר מהר. אבל אם המדרון הוא אפס, המודל מפסיק ללמוד. להסביר את זה בצורה מתמטית, gradient הוא נגזרת חלקית ביחס ל Input שלה.

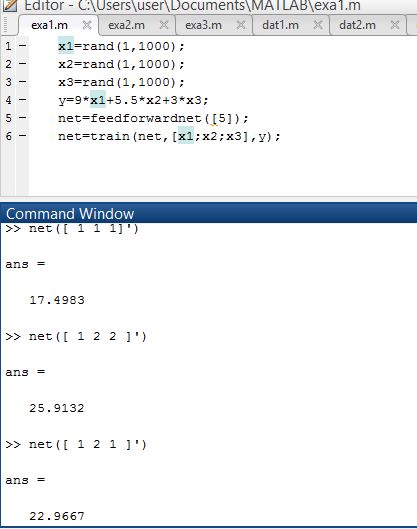
**דוגמאות מה MATLAB :**

(1 דוגמה רגציטיה לינארית:

**y**, **x1**, **x2**, **x3** הם הנתונים שעל גרף בפונקציה: **y**=9**X1**+5.5**X2**+3**X3**

המחשב לא מכיר את בפונקציה, אז בעזרת הnet המחשב מגלה את בפונקציה. אחר כך המחשב מתאמן על הנתונים כדי למצוא את הפתרון לפונקציה (ה Y).

אחרי שהוא מתאמן על הנתונים, אם נתן ערך ל **x1**, **x2**, **x3**הוא מוצא את **Y**.

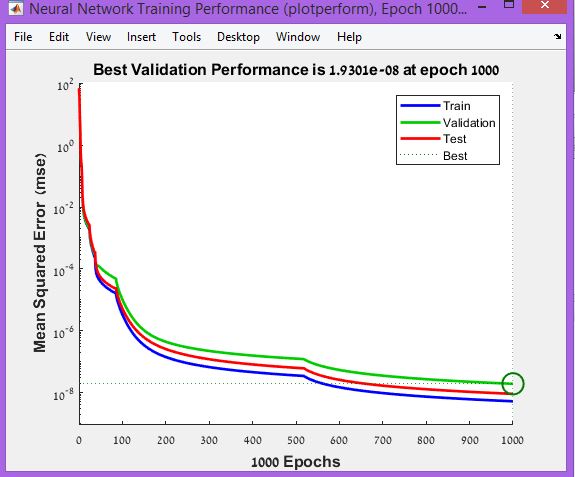
****

השרות 1-3 מגדירות 3 ויקטורם מ1000 תא בין 0-1 (הנתונים שה "net" יקבל).

השורה 4 מגדירה את הפונקציה שאנו רוצים (התוצאה).

השורות 5-6 מגדירות רשת ומלמדים אותה שאם יתנו לה **1X 2X 3X** שתוציא **Y** .

**גרף שמראה איך ה performance ירד במהלך הלימידה :**



**2) דוגמה לא לינארית:**

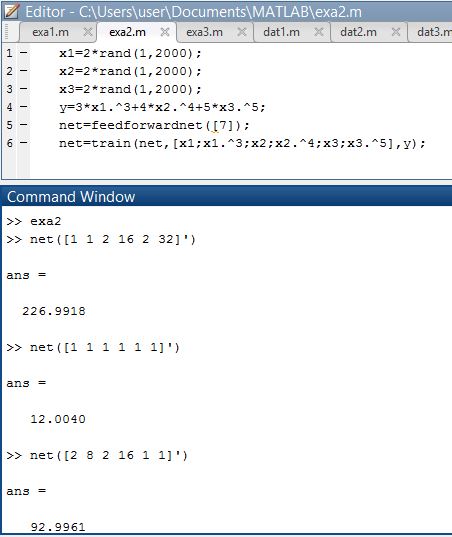
**א-**

**y**, **x1**, **x2**, **x3** הם הנתונים שעל גרף בפונקציה: **y**=3**X1**^3+4**X2**^4+5**X3**^5

המחשב לא מכיר את בפונקציה, אז בעזרת הnet המחשב מגלה את בפונקציה. אחר כך המחשב מתאמן על הנתונים כדי למצוא את הפתרון לפונקציה (ה Y).

אחרי שהוא מתאמן על הנתונים, אם נתן ערך ל **x1**, **x2**, **x3**הוא מוצא את **Y**.

כדי לעזור למחשב לפתור את המשוואה אנחנו מכניסים לפונקציה X בחזקה.

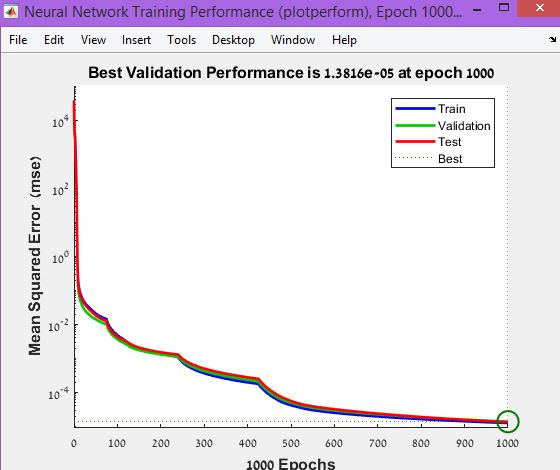


השרות 1-3 מגדירות 3 ויקטורם מ2000 תא בין 0-2 (הנתונים שה "net" יקבל)

השורה 4 מגדירה את הפונקציה שאנו רוצים (התוצאה)

השורות 5-6 מגדירות רשת ומלמדות אותה שאם יכניסו בה נתונים מתאמים תוציא את Y ובמצב הזה **1X 2X 3X** הם (x2^4 x1^3 X3^5) כדי שתצא לנו פונקציה יותר מדויקת מומלץ להכניס את X בחזקה למשוואה.

**גרף שמראה איך ה performance ירד במהלך הלימידה :**



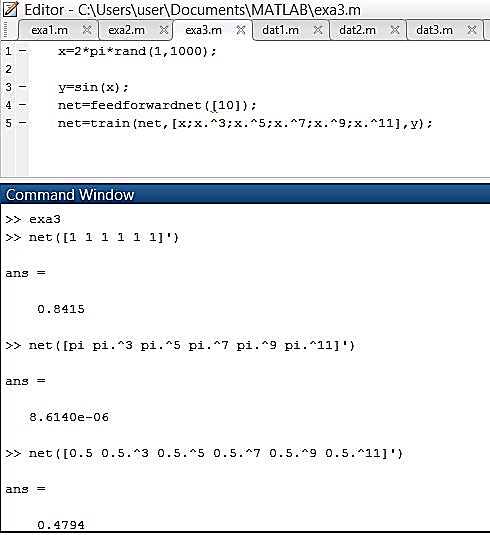
**ב-**

**y**, **x** הם הנתונים שעל גרף בפונקציה: **y**=sin(**x**)

המחשב לא מכיר את בפונקציה, אז בעזרת הnet המחשב מגלה את בפונקציה. אחר כך המחשב מתאמן על הנתונים כדי למצוא את הפתרון לפונקציה (ה Y).

אחרי שהוא מתאמן על הנתונים, אם נתן ערך ל **x** הוא מוצא את **Y**.

כדי לעזור למחשב לפתור את המשוואה אנחנו מכניסים לפונקציה X בחזקה.



השרות 1-3 מגדירות 3 ויקטורם מ1000 תא בין π0-2 (הנתונים שה "net" יקבל)

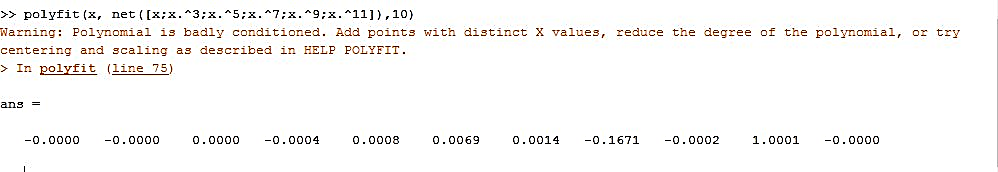
השורה 4 מגדירה את הפונקציה שאנו רוצים (התוצאה)

השורות 5-6 מגדירות רשת ומלמדות אותה שאם נכניס בה נתונים מתאמים היא תוציא Y ובמצב הזה ה(input)x הוא (x^3 x^5 x^7 x^9 x^11) בגלל ש:

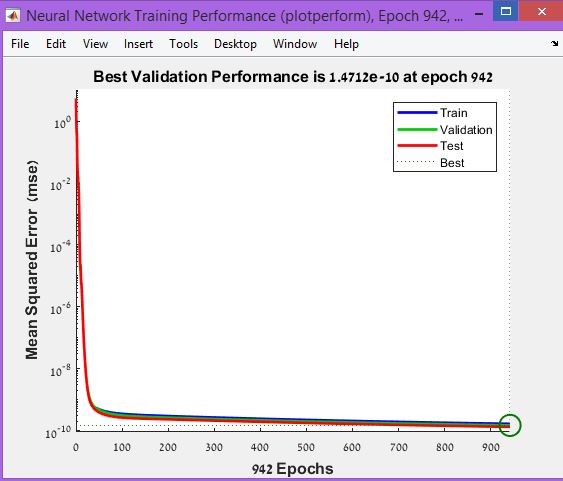
אז כדי שתצא לנו פונקציה יותר מדויקת מומלץ להכניס X בחזקה לתוך המשוואה.

**וכך מצאנו את המקדמים של X לפונקציית סינוס בטור טיילור.**

**וגם מצאנו את ארבעת המקדמים הראשונים בדיוק :**

****

**גרף שמראה איך ה performance ירד במהלך הלימידה :**



**תיאור המודל:**

נושא העבודה שלנו הוא למידת ששה סוגים של סדרות, בזה שאנחנו מכניסים לרשת נתונים ונותנים לה להתאמן עליהם כך שאנחנו

ולזהות אותם מתוך בניית אלגוריתם שמכניסים לו data שכוללת תמונות רבות של פרצופי אנשים עם הרגש שהתמונה מביעה כלומר כעס-עצב-שמחה ועוד שסדרנו במספרים מאפס עד שש. המטרה של העבודה היא למצוא דרכים שבה המודל מזהה את הפרצופים כלומר את הרגשים שהתמונות מביעות בצורה הכי מדויקת וטובה וזאת היא ה"בעיה" שאנחנו מנסים לפתור.

**סיכום:**

מהעבודה הזאת למדנו על נושא חדש שהוא אנטלגנציה מלאכותית ואיך לבנות רשת ניורונים, היה מעניין ללמוד איך מאמנים רשת שלא רק מזהה את סוג הסדרה מהנתונים שהאכלנו אותה, אבל גם מוציאה את האיבר הבא בסדרה זאת.

בנוסף לזה, העבודה הזאת שמה אותנו במצב מאתגר והיינו צריכים ללמוד איך לעשות עבודה שהיא לא קלה ואיך להתמודד עם האתגר של לפתור אותה.