בס"ד

27.04.2017

סמסטר נירו חישוביים
ב' – 2017
מגישים:
יקר נדב
אפראימוב אורן

סביבת עבודה:
התרגיל הורץ ונבדק על מערכת Ubuntu ומערכת windows 10.
במערכת windows התרגיל הורץ על גבי פייתון גרסת 3.5.2 אנקודה (anaconda python). במערכת Ubuntu התרגיל הורץ על גבי פייתון גרסת 3.5.

התרגיל נכתב בסביבת פיתוח Pycharm.

תיאור האלגוריתם – ההנחות ומימוש של BackPropagation:
האלגוריתם מבוסס על BackPropagation one-by-one, כל אפוק משוערכת המערכת אל מול הטסט סט המתאים ומחושבת הטעות באמצעות Soft max עם חסם תחתון של 0.8.
במהלך הריצה מדווחים בזמן אמת אחוזי הטעות של כל אפוק וזמני הריצה שלו.
תנאי העצירה הינו לריצה על פולד מסויים כאשר הגיע למגבלת האפוקים כפי שנתונה בשורת הפקודה או כאשר ב3 בדיקות רצופות מזהים שלא היה שיפור גדול מ-0.001 בין אפוק אחד לבא אחריו. בסיומה, אנו מדווחים על אחוז הטעות הממוצע והשונות על פני על הפולדים, וכן את זמן הריצה הכולל. כדי להפעיל את התכנית יש להריצה דרך ה-command line, ולספק את מספר השכבות, מספר נוירונים בכל שכבה, וכו'. הפעלת התכנית ללא פרמטרים תפרט את הפרמטרים הנדרשים וכן, בסוף הפירוט, דוגמאות לשורת ההרצה כדי להקל על הבדיקה.
ישנם שני מצבי ריצה אפשריים – יינות או אותיות, כאשר כל מצב יקבל קלטים שונים – היינות את שני תיקיות היינות, ולאחר מכן יסווגו בהתאם ויוסרו כפילויות מידע. במצב ה'אותיות' תקבל התכנית הפניה לתיקיה המכילה תתי תיקיות בשמות האותיות (קונבנציה) – הרשימה מפורטת מפירוט הניתן מההרצה ללא פרמטרים של שורת הקלט

חלק א' – סיווג יין אדום ולבן
ניתוח המידע – לפני ההרצה
קיימים 2 סוגים שונים של יין שכול אחד יושב בקובץ נפרד. שני הקבצים מכילים את אותם פיצ'רים (11 פיצ'רים שמרכיבים יחד את ציון quality של אותו יין (כל שורה מציינת בדיקה אחרת)
אנו שמים לב לכמה נקודות חשובות:

- טווח המספרים ש-Quality יכול להיות:

בס"ד

27.04.2017

- יין אדום – טווח המספרים הוא בין 3-8 של מספרים שלמים (כולל הקצוות)
- יין לבן – טווח המספרים הוא בין 3-9 של מספרים שלמים כולל הקצוות)

- כמות הנתונים:
  - יין אדום – מכיל 4898 דוגמאות, שמתוכם רק 3961 הם <u>דוגמאות שונות</u>!
  - יין לבן – מכיל 1599 דוגמאות, שמתוכם רק 1359 הם <u>דוגמאות שונות</u>!

בס"ד

27.04.2017

סטטיסטיקה – לפני ההרצה

כמו שהראנו בסעיף הקודם, הנתונים מאוד לא מייצגים את העולם האמתי!
נראה סטטיסטיקות שעשינו על הנתונים על מנת לעמוד את טיב המודל שבנינו.
נשים לב, בחלק מן הארכיטקטורות, קיבלנו תוצאות טובות ביחס לשאר היות והתפלגות הנתונים לא אחיד.

1. כפילויות:
כמו שהראנו בסעיף הקודם, קיים כפילויות במידע שקיבלנו שיכולים להשפיע על תהליך הלמידה, היות והם יכולים להיכנס לחלק של test set וכך נעשה בדיקה על מידע שעליו אמינו את המערכת (מידע עברו נקבל זיהוי נכון היות והמערכת כבר התאמנה עליו)

התפלגות לפי אחוזים:

- יין לבן - $\frac{937}{4898} \approx 0.19$, כלומר כמעט 20 אחוז ממידע הוא מידע כפול שבסבירות די גבוהה נקבל את אותו example גם בtest set-
- יין אדום - $\frac{240}{1599} \approx 0.15$, כלומר 15 אחוז ממידע הוא מידע כפול שבסבירות די גבוהה נקבל את אותו example גם בtest set-.

כמות examples עובר כל label:
ללא כפילויות:

- יין לבן - עבור התווית 3 קיים 20 דוגמאות, עבור התווית 4 קיים 163 דוגמאות, עבור התווית 5 קיים 1457 דוגמאות, עבור התווית 6 קיימים 2198, עבור התווית 7 קיימים 880 דוגמאות, עובר התווית 8 קיימים 175 דוגמאות ועבור התווית 9 קיימים 5 דוגמאות.
Baseline – $\frac{2198}{4898} \approx 0.448$ במידה ונחזיר את ורק את התווית מספר 6.

- יין אדום – עבור התווית 3 קיים 10 דוגמאות, עבור התווית 4 קיים 53 דוגמאות, עבור התווית 5 קיים 681 דוגמאות, עבור התווית 6 631, עבור התווית 7 קיימים 199 דוגמאות ועובר התווית 8 קיימים 18 דוגמאות
Baseline – $\frac{681}{1599} \approx 0.425$ במידה ונחזיר את ורק את התווית מספר 5.

בס"ד

27.04.2017

<u>עם כפילויות:</u>

- יין לבן - עבור התווית 3 קיים 20 דוגמאות (**אין** דוגמאות כפולות), עבור התווית 4 קיים 153 דוגמאות (**10** דוגמאות כפולות), עבור התווית 5 קיים 1175 דוגמאות (**282** דוגמאות כפולות), עבור התווית 6 קיימים 1788 (**410** דוגמאות כפולות), עבור התווית 7 קיימים 880 דוגמאות (**191** דוגמאות כפולות), עובר התווית 8 קיימים 175 דוגמאות (**44** דוגמאות כפולות) ועבור התווית 9 קיימים 5 דוגמאות (**אין** דוגמאות כפולות).
  <u>Baseline</u> – $\frac{1788}{3961} \approx 0.451$ במידה ונחזיר את ורק את התווית מספר 6.

- יין אדום – עבור התווית 3 קיים 10 דוגמאות (**אין** דוגמאות כפולות), עבור התווית 4 קיים 53 דוגמאות (**אין** דוגמאות כפולות), עבור התווית 5 קיים 577 דוגמאות, (**104** דוגמאות כפולות), עבור התווית 6 535 (**94** דוגמאות כפולות), עבור התווית 7 קיימים 167 דוגמאות (**32** דוגמאות כפולות) ועובר התווית 8 קיימים 17 דוגמאות (**דוגמא** אחד כפולה).
  <u>Baseline</u> – $\frac{577}{1359} \approx 0.424$ במידה ונחזיר את ורק את התווית מספר 5.

בס"ד

27.04.2017

אנליזה ובחירת ארכיטקטורה:

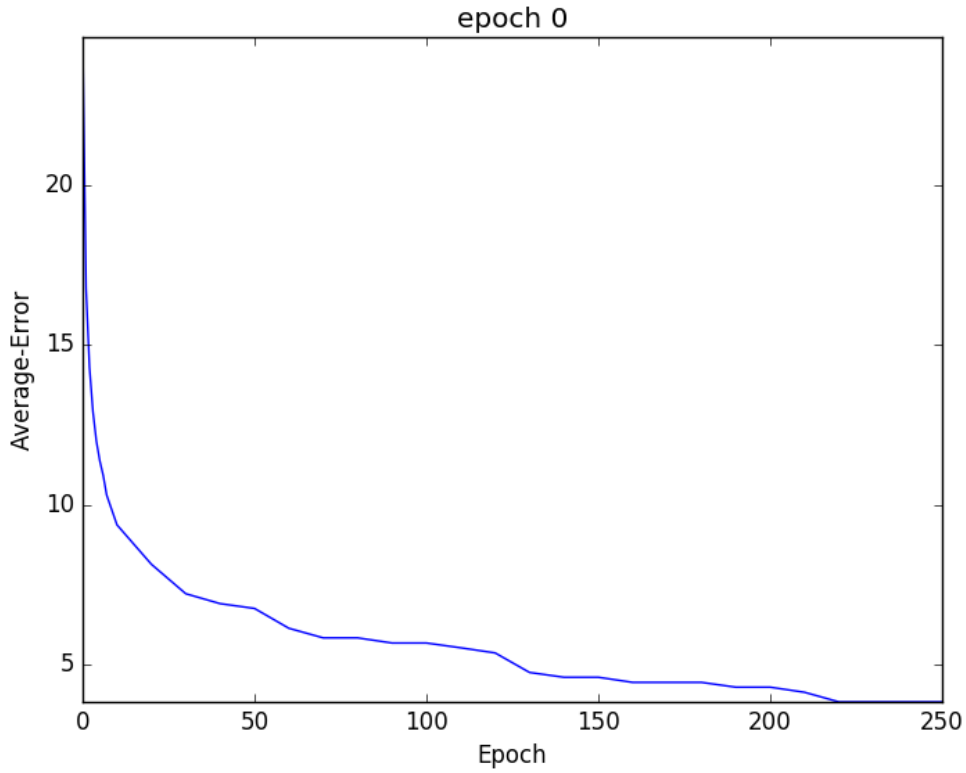
בחרנו לממש 4 מודלים עם כמות hidden layers שונה על מנת לעמוד את טיב המודל שיצרנו. חשוב לזכור, היות ומודבר ב"סיווג" בין יין אדום ליין לבן אנו צריכים לממש מערכת שיודעת לסווג יותר טוב מ-**baseline** שהוא $\frac{4898}{6497} \approx 0.753$ עבור מודל שלא מוריד את הכפילויות במידע או $\frac{3961}{5320} \approx 0.744$ עבור מודל שמוריד כפילויות במידע.

מודל מספר 1

המודל הראשון שבחרנו לנתח מורכב מ:

- אלפא – 0.3.
- מספר איטרציות מקסימלי (epochs) – 1000 (המודל עוצר לאחר שהוא לא רואה שינוי ב-3 האטרקציות האחרונות).
- רמה בודדת של נוירונים:
  - רמת הפלט – מכילה 2 נוירונים.

תוצאת:



בס"ד

27.04.2017

ניתוח זמן ריצה למודל בודד:

Time	error	epochs
1.39s	24.62	0
1.62s	16.77	1
1.15s	14.31	2
1.13s	12.92	3
1.87s	12	4
2.59s	11.38	5
1.24s	10.92	6
…	….	…
15.9	4.31	190
19.5s	4.31	200
22.73s	4.15	210
16.52	3.85	220
16.22s	3.85	230
17.15s	3.85	240
15.2s	3.85	250

average error: 4.23%
average deviation over the folds: 0.09
learning time for all folds: 34:21

מודל מספר 2

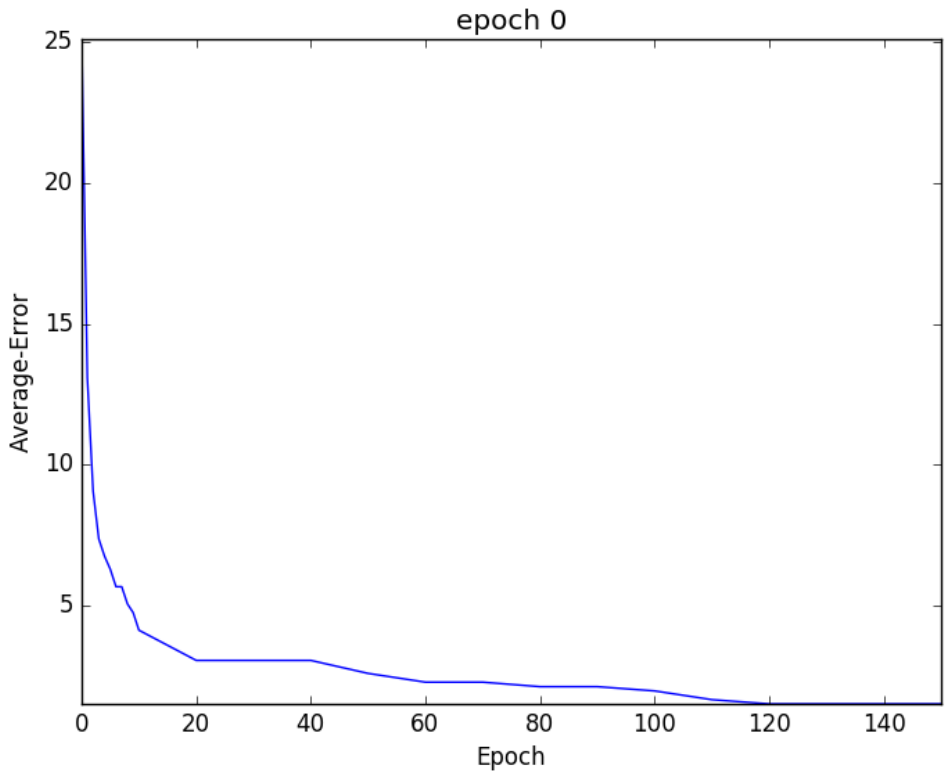
המודל הראשון שבחרנו לנתח מורכב מ:

- אלפא – 0.3.
- מספר איטרציות מקסימלי (epochs) – 1000 (המודל עוצר לאחר שהוא לא רואה שינוי ב-3 האטרקציות האחרונות).
- 2 רמות נוירונים:
  - רמת הקלט – מכילה 6 נוירונים.
  - רמת הפלט – מכילה 2 נוירונים.

בס"ד

27.04.2017

תוצאת:



ניתוח זמן ריצה למודל בודד:

Time	error	epochs
	25.08	0
	13.08	1
	9.08	2
	7.38	3
	6.77	4
	6.31	5
	5.69	6
…	….	…
	2.15	80
	2.15	90
	2	100
	1.69	110
	1.54	120
	3.85	130
	3.85	140

בס"ד

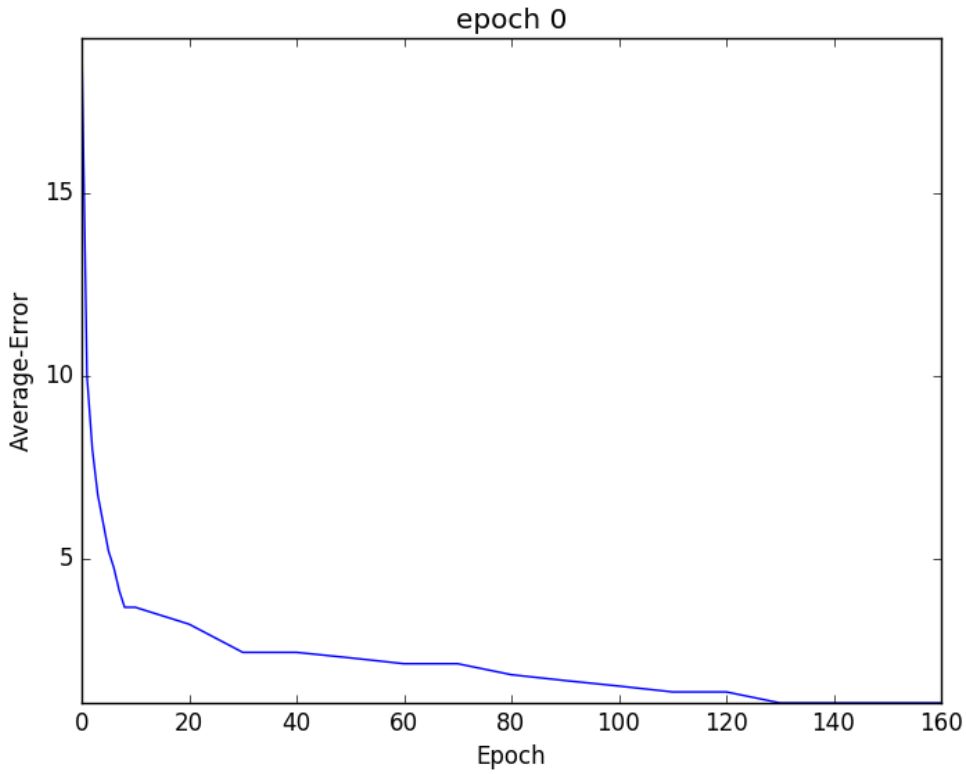
27.04.2017

מודל מספר 3

המודל הראשון שבחרנו לנתח מורכב מ:

- אלפא – 0.3.
- מספר איטרציות מקסימלי (epochs) – 1000 (המודל עוצר לאחר שהוא לא רואה שינוי ב-3 האטרקציות האחרונות).
- 2 רמות נוירונים:
  - רמת הקלט – מכילה 20 נוירונים.
  - רמת הפלט – מכילה 2 נוירונים.

תוצאת:



בס"ד

27.04.2017

ניתוח זמן ריצה למודל בודד:

time	error	Epochs
6.28s	19.23	0
5.46s	10	1
5.56s	8	2
5.67s	6.77	3
5.14s	6	4
5.84s	5.23	5
6.06s	4.77	6
…	….	…
70.69	1.54	100
64.38s	1.38	110
61.48s	1.38	120
74.22s	1.08	130
76.32s	1.08	140
74.52s	1.08	150
75.19s	1.08	160

average error: 21.28%
average deviation: 0.06
learning time for all folds: 392:33

מודל מספר 4

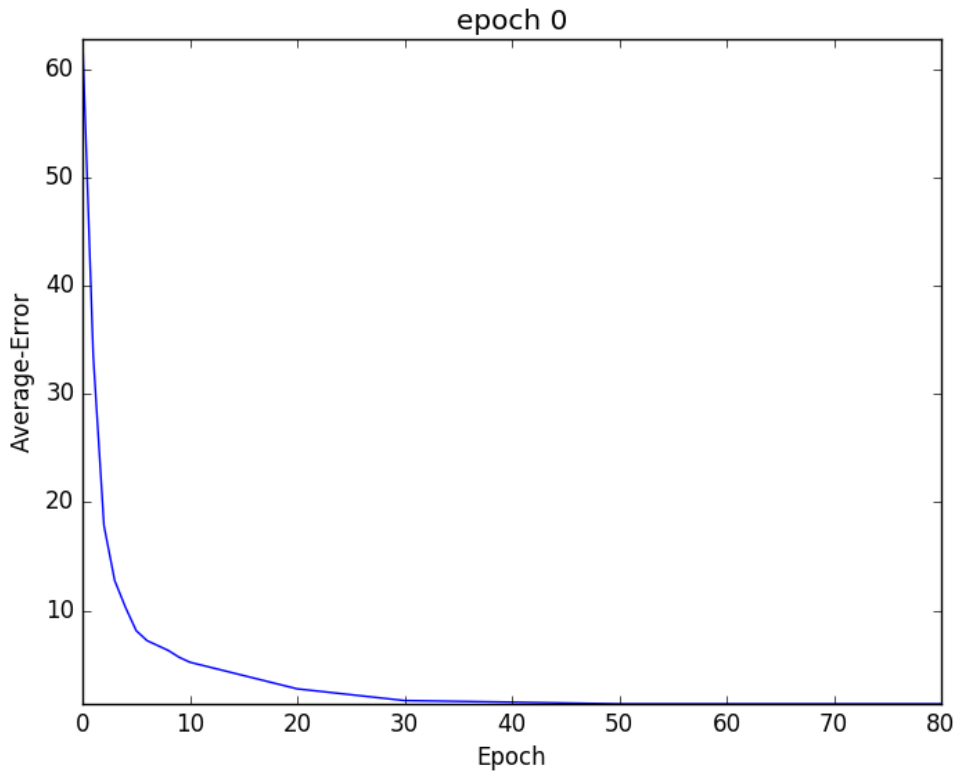
המודל הראשון שבחרנו לנתח מורכב מ:

- אלפא – 0.3.
- מספר איטרציות מקסימלי (epochs) – 1000 (המודל עוצר לאחר שהוא לא רואה שינוי ב-3 האטרקציות האחרונות).
- 3 רמות נוירונים:
  - רמת הקלט – מכילה 20 נוירונים.
  - רמת hidden – מכילה 6 נוירונים.
  - רמת הפלט – מכילה 2 נוירונים.

בס"ד

27.04.2017

תוצאת:



ניתוח זמן ריצה למודל בודד:

time	error	Epochs
4.84s	62.77	0
3.95s	33.54	1
4.96s	17.85	2
4.22s	12.77	3
4.46s	10.31	4
4.15s	8.15	5
…	….	…
64.38s	1.54	40
61.48s	1.38	50
74.22s	1.38	60
76.32s	1.38	70
74.52s	1.38	80
75.19s	1.38	90

average error: 1.38%
average deviation over the folds: 0.00
learning time for all folds: 148:46

בס"ד

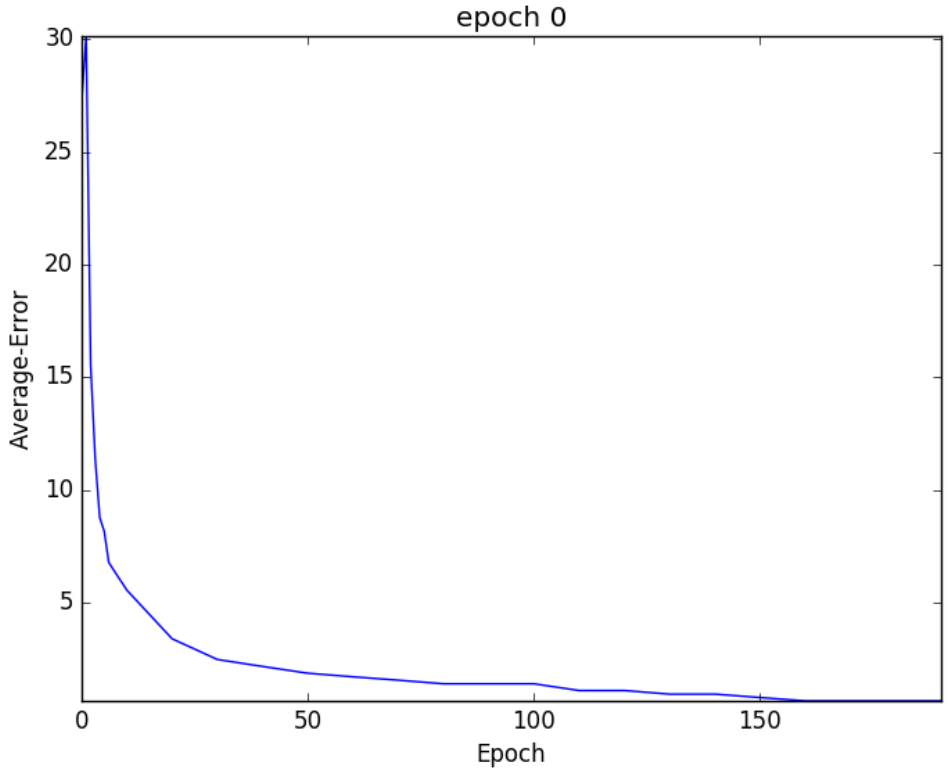
27.04.2017

מודל מספר 5

המודל הראשון שבחרנו לנתח מורכב מ:

- אלפא – 0.3.
- מספר איטרציות מקסימלי (epochs) – 1000 (המודל עוצר לאחר שהוא לא רואה שינוי ב-3 האטרקציות האחרונות).
- 3 רמות נוירונים:
  - רמת הקלט – מכילה 6 נוירונים.
  - רמת hidden – מכילה 20 נוירונים.
  - רמת הפלט – מכילה 2 נוירונים.

תוצאת:



ניתוח זמן ריצה למודל בודד:

time	error	Epochs
	27.23	0
	30.15	1
	15.54	2
	11.38	3
	8.77	4
	8.15	5
	6.77	6
…	….	…
	1.08	120
	0.92	130
	0.92	140

בס"ד

27.04.2017

140	0.77
150	0.62
160	0.62
170	0.62

<u>ניתוח ואנליזה</u>
הכנו 5 מדולים שכול מודל מכיל מספר שונה של נוירונים ושכבות.
הכנו מודל יחיד בעל שכבה בודדת (מודל מספר 1), 2 מודלים בעלי 2 שכבות (מודלים מספר 2 ו-3 בהתאמה) ו-2 מודלים בעלי 3 שכבות (מודלים 4 ו-5 בהאמתה).

ראשית, נזכור שמדובר בסיווג לינארי, לכן צפינו שכול המודלים יתנו תוצאות טובות, אך לא צפינו לרמות כאלו של דיוק.
נזכור ש-baseline הוא 75.3 אחוז זיהוי ולכן, צפינו שמודל הכי גרוע שנבחר יהיה סביבות 80-90 אחוזי זיהוי.
נשים לב, שכל המודלים שבחרנו התכנסו די מהר, מודל 2 אחרי 120 אפוקים (אחוז טעות של 1.38% - בעל 2 שכבות) עד המודל האיטי ביותר מודל 1 אחרי 250 אפוקים (אחוז טעות של 3.85% - בעל שכבה בודדת).

דבר אחרון, נשים לב ככול שעולים בכוחו של המודל כן גם אחוז הטעות קטנה, אנו חושבים שזה בגלל שאנחנו מסווגים בעיה פשוטה בעזרת יותר פרמטרים וכך מצליחים להתכנס מהר יותר לנקודת מינימום. באופן מפתיע, מודל 4 בעל 3 שכבות, 20 נוירונים בקלט, 6 ברמה הנסתרת ו-2 בפלט המודל התכנס די מהר ואחרי 80 אפוקים הגענו לאחוז טעות של 1.38%.

<u>המודל עם הזיהוי הגדול ביותר</u> היה מודל מספר 5 (בעל 3 השכבות), אשר התחיל עם אחוז טעות שך 27.23% ואחרי 190 אפוקים הגיע ל-0.62.
<u>המודל עם אחוז הזיהוי "הקטן" ביותר</u> היה מודל מספר 1 (בעל שכבה בודדת).

<u>סיכום</u>
אחרי ניתוח המידע, גילנו שמודל עם יותר שכבות מתכנס יותר טוב לבעיה לינארית. המודלים בעלי 2 ו-3 רמות התכנסו יותר מהר, ואנו חושדים שזה בגלל כמות הפרמטרים שיש בידי המערכת. אחרי מעט אפוקים הם מתקרבים מהר מאוד לנקודת המינימום וכך צריכים פחות תיקונים של המשקלים.
מנגד, באופן לא מפתיע, מודל בעל שכבה אחד התכנס בקצב הכי איטי, וזאת בגלל שהוא בעל כמות פרמטרים לא גדולה ולכן היה צריך לעבור יותר אפוקים על מנת להגיע לנקודת המינימום. מודל בעל שכבה בודדת, צריך יותר אטרקציות על מנת לעדכן את המשקלים וכך להגיע לנקודת מינימום.

בס"ד

27.04.2017

חלוקה ל-3 רמות

לדעתנו, קשה יהיה לחלק את המידע ל-3 רמות מסיבות הבאות:

- מספר הדוגמאות שונה – מספר הדוגמאות של בקובץ של יין לבן גדול פי 3 ממספר הדוגמאות בקובץ אדום.
- Quality אינו תואם – ביין הלבן קיים quality label שאינו נמצא ביין אדום.
- Label דומיננטי – בכול אחד מן הקבצים קיים label דומיננטי. בקובץ של היין האדום זה quality מספר 5 ואילו בקובץ הלבן זה quality מספר 6.

לכן, כל חלוקה אשר נבחר לא תהיה טובה עקב הסיבות שפירטנו.
יחד עם זאת, יהיה ניתן לסדר את כמות ה-labels כך שיהיה תואם, ולחלק את הנתונים ל-3 רמות שהם:

- Quality 5 – הlabel הראשון.
- Quality 6 – הlabel השני.
- All the rest – איחוד כל שאר התגיות.

לדעתנו, חלוקה זו תהיה מיטבית עם מספר הדוגמאות היה באותו סדר גודל, עקב כך שיש פער ענק בין שני הקבצים, פי 3, אנו חושבים שנתונים ייטעו לכיוון ה-label עם כמות הדוגמאות הרבה ביותר (במקרה שלנו תווית מספר שתכיל 2198 דוגמאות מיין הלבן + 631 דוגמאות מיין האדום – 2829 דוגמאות שמהווה 43.5 אחוז מdataset).

בסה"כ קיבלנו קובץ שדומה מאוד ל-2 הקבצים (באופן "לא" מפתיע").
התוצאות יהיו מאוד דומות במידה ונעשה את אותו חלוקה עבור כל קובץ בנפרד, כי הם שומרים על אותה ההתפלגות של המידע.

בס"ד

27.04.2017

חלק ב' – תמונות

רקע:
על מנת להפוך את התמונות שייצרנו לקובץ בינארי (התמונות שנמצאים בתיקייה Part 2 - images input), בנינו את המחלקה ImageClass. הגדרנו פונקציה אשר עוברת על כל התיקייה שם נמצאים התמונות (Part 2 - images input) וממירה כל תמונה לקובץ בינארי. תיקיית הפלט היא Part 2 - images output.

ניתן לשנות את שמות התיקיות וכך ניתן לשמור את כל הקבצים שנוצרים תחת שמות תיקיות אחרות.

שם הפונקציה – convert_image_to_binary_file
images input – מכילה את התמונות שיצרנו, 4 עבור כל אות.
images output – מכילה את הקבצים הבינאריים עבור כל אות שיצרנו.

מניפולציות על התמונות
על מנת להגדיל את מאגר הנתונים, אשר מכיל 14 תיקיות (כל זוג הכין תיקייה עם 32 תמונות, 4 עבור כל אות), הגדרנו סט של פונקציות אשר מטרתם להגדיל את סט התמונות.
עבור כל תמונה, הגדרנו את סט המניפלציות הבא:

- Shift 3 – עבור כל תמונה הזזנו אותה (במידה וזה אפשרי) 3 פיקסלים לכול צד: ימין, שמואל, למעלה ולמטה.
- Noise - עבור כל תמונה מקורית (יחד עם התמונות שעברו shift), הגרלו באופן רנדומלי עבור כל שורה את העמודה בה נעשה את הרעש. ביצענו xor על ערך הקיים בתא.
- Large noise – כמו קודם, רק שפה ניתן לבחור את גודל הרעש, אנחנו הגדרנו שברירת המחדל שכל שורה תכיל 2 רעשית (סה"כ הוספנו רעש למערכת בגודל של 12.5 $\frac{32}{256} = 12.5$).

שם הפונקציה – manipultated_on_images.
Images – Input – מכיל את כל התיקיות של הקבצים הבינארים שכול זוג סטודנטים הכינו.
Images – Dataset – מכיל את התמונות אחרי שעברו מניפולציה מסודרים בתיקייה אליה הם שייכים.

חשוב לזכור, הפונקציה manipultated_on_images, מציגה על גבי המסך (עבור אות אחד) את פעולות shift ורעש. בדרך כלל לאחר 12

בס"ד

27.04.2017

תמונות, החלון יורד ותוכנית ממשיכה לייצר את התמונות ללא הצגה למשתמש.

<u>תיאור הנתונים:</u>
סט הנתונים הרגיל מכיל 447 תמונות, שלאחר מניפולציות (הזזה, רעש רעש "גדול" יותר) הגנו ל-4023 תמונות. כל תמונה עברה כל אחד ממניפולציות שהגדרנו קודם, 4 תמונות (shift לכול כיוון), עבור כל תמונה שעברה shift יחד עם התמונה המקורית עשינו עליהם רעש (5 תמונות נוספות) ואותו דבר עשינו עבור רעש גדול יותר (5 תמונות נוספות).

סה"כ: כל תמונה שוכפלה ל-15 תמונות נוספות (בהנחה וכל shift הצליחו).

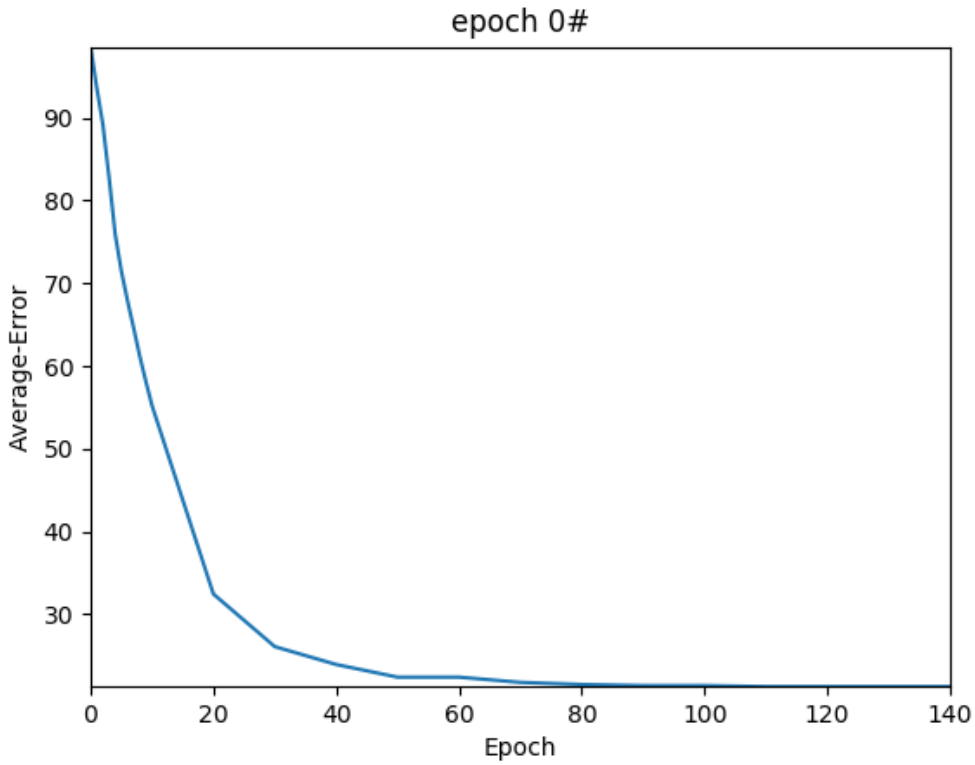
<u>מודל מספר 1</u>
המודל הראשון שבחרנו לנתח מורכב מ:

- אלפא – 0.3.
- מספר אטרקציות מקסימלי (epochs) – 1000 (המודל עוצר לאחר שהוא לא רואה שינוי ב-3 האטרקציות האחרונות).
- 2 רמות נוירונים:
  - רמת הקלט – מכילה 20 נוירונים.
  - רמת הפלט – מכילה 8 נוירונים.

בס"ד

27.04.2017

תוצאת:



ניתוח זמן ריצה למודל בודד:

time	error	Epochs
	98.60	0
	93.64	1
	89.19	2
	82.82	3
	75.95	4
	71.50	5
	67.94	6
…	….	…
	22.39	60
	21.76	70
	21.50	80
	21.37	90
	21.37	100
	21.25	12
	21.25	130
ש 1.625	21.25	140

בשונה מחלק הראשון של התרגיל, כן אחוזי השגיאה של האפוק הראשון מאוד גדול (98.6 אחוז שגיאה). שחושבים על זה, זה מאוד הגיוני, כי בחלק הזה של התרגיל, אנו נותנים למודל להחליט מבין 8 אפשרויות, בשונה מחלק הראשון ששם זה היה בחירה בין יין לבן או אדום (0 או 1).
לאחר כל אפוק, אחוז השגיאה יורד עד שאנו מגיעים למינימום מקומי. אחרי 20 אפוקים שלקחו 40 דקות הגענו ל32.44 אחוז שגיאה, ולאחר עוד 10 אפוקים אנחנו כבר אנחנו כבר כמעט מגיעים למינימום המקומי.
נקודה חשובה שצריך לתת לה דגש, אחרי 30 אפוקים ועד 140 המודל שיפר את התוצאות שלו במעט שזה מראה שארכיטקטורה שנבחרה וסט ההנחות (המודל, הפונקציה וטווחים של בחירת המשקל עובר כל נוירון) היו מאוד מאוד טובים.

סה"כ: הרצת כל התרגיל, 140 אפוקים לקחו באיזור ה-1.65 שעות, וזה בגלל שהיו למודל המון דוגמאות וכל עדכון משקלים לקח זמן מאוד גדול.

<u>תוצאות ושונות:</u>

average error: 21.28%
average deviation: 0.06
learning time for all folds: 392:33

נשים לב, שעבור כל אחד מ-(kfold=4) fold, אנו מקבלים שכול אחד מן המודלים מגיע לאותו אחוז שגיאה ושונות היא אך ורק 0.06.
כלומר, כל אחד מן הריצות הגיע לאותו אחוז שגיאה, דבר שמראה שתמונות שבחרנו אכן מייצגת וכנראה הגענו למינימום הגלובאלי של המערכת.

<u>מודל מספר 1</u>

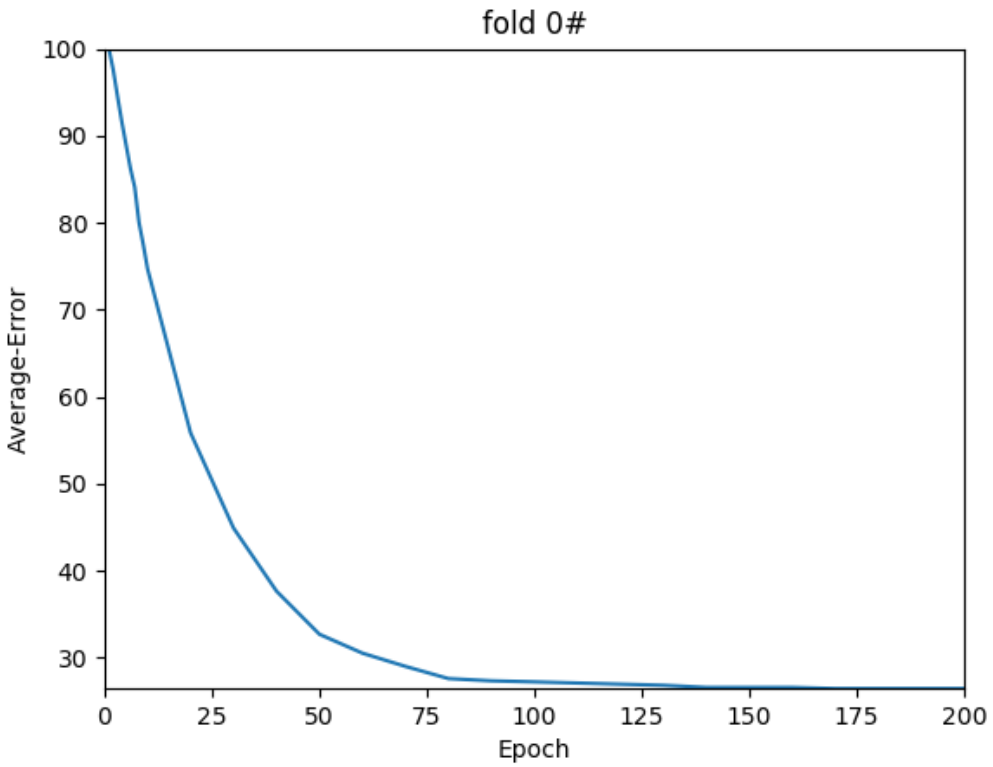
המודל הראשון שבחרנו לנתח מורכב מ:

- אלפא – 0.3.
- מספר איטרציות מקסימלי (epochs) – 1000 (המודל עוצר לאחר שהוא לא רואה שינוי ב-3 האטרקציות האחרונות).
- 2 רמות נוירונים:
  - רמת הקלט – מכילה 20 נוירונים.
  - רמת הפלט – מכילה 8 נוירונים.

בס"ד

27.04.2017

תוצאת:



ניתוח זמן ריצה למודל בודד:

time	error	Epochs
	27.23	0
	30.15	1
	15.54	2
	11.38	3
	8.77	4
	8.15	5
	6.77	6
…	….	…
		20
…	..	…
	21.37	90
	21.37	100
	21.25	110
	21.25	120
	21.25	130
	21.25	140

בס"ד

27.04.2017

בשונה מהרצה הקודמת, קצב התכנסות המודל כאן יותר איטית. אנו חודשים שזה יכול לנבוע בגלל שמודל קיבל בהתחלה תמונות שלא עזרו לו לתקן נכון את המשקלים, כלומר הוא ראה דוגמאות של אותיות שהוא כבר ראה קודם ולכן זה לא עזר לו להתקדם לכיוון המינימום. לאחר 60 אפוקים (במודל הראשון זה היה 30 אפוקים) אנו מקבלים את אותו אחוז שגיאה כמו שקיבלנו במודל הראשון. בדומה למודל הקודם, גם כאן לקח למודל המון זמן להתכנס למינימום.

בסך הכול, בדומה לקודם, גם כאן המודל הגיע לאחוז שגיאה די טוב, רק 26.64% סיווגים לא נכונים וזה מחזק את העבודה שסט הנחות ובניית המודל שלנו הייתה די טובה.

סה"כ: רצית כל התרגיל, 140 אפוקים לקחו באיזור השעתיים ו-20 דקות, וזה בגלל שהיו למודל המון דוגמאות וכל עדכון משקלים לקח זמן מאוד גדול.

<u>תוצאות ושונות:</u>

average error: 26.49%
average deviation: 0.19
learning time for all folds: 556:43

נשים לב, שעבור כל אחד מ-(kfold=4) fold, אנו מקבלים שכול אחד מן המודלים מגיע לאותו אחוז שגיאה ושונות היא אך ורק 0.19.

אם ניקח בחשבון גם את התוצאות של המודל הקודם, נשים לב שמערכת מגיע לאותם תוצאות וכנראה הגענו למינימום מקומי.
בשני המודלים השונות היו די קטנים, וזה מראה או שנתונים שבחרנו זהים בכול אחד מן הfold (התפלגות העולם זהה) או שמודל שלנו בנוי היטב ולכן אנו מגיעים לאחוזי שונות קטנים מאוד.