**Multilayer Perceptron**

הפרויקט עוסק בזיהוי מגדר לפי קטעי קול שהוקלטו והומרו למאפיינים ווקליים. זיהוי המגדר נעשה ע"י Multilayer Perceptron שממומש בעזרת Tensorflow.

הפרויקט חולק לארבעה חלקים כאשר בין החלק השני לשלישי הגדרנו ובנינו את מודל רשת הנויירונים שלנו.

**חלק ראשון(read\_data\_set):**

החלק הראשון עוסק בקריאת ה data set של המודל (במודל שלנו הוא מוצג כקובץ csv) על ידי שימוש בספרייה pandas. בגלל שהפרמטר label במודל מוצג כערכים male, female אז הוחלפו הערכים שיהיו 0,1 בהתאמה ואז עורבבו השורות על מנת שה-dataset לא יהיה ממויין (בקובץ csv מופיעים קודם כל הדגימות של הגברים ורק לאחר מכן הדגימות של הנשים) ואז המידע הוצב במשתנה voice.

**חלק שני(split\_data\_set):**

החלק השני עוסק בלקיחת המידע שהושג בחלק הראשון וחילוקו ל train ו- test. החילוק נעשה על ידי הכנסת הנתונים שנמצאים במשתנה voice לתוך שני מערכים מהספרייה numpy. מערך ראשון data\_y מכיל את כל הערכים שנמצאים בפרמטר label ומערך שני data\_x מכיל את שאר הערכים שלא הוכנסו מקודם. אחרי החילוק של שני המערכים כל מערך חולק לשני מערכים ביחס של 70-30. ארבעת המערכים הוצבו לתוך המשתנים train\_data\_x , train\_data\_y , test\_data\_x , test\_data\_y.

בין החלק השני לשלישי הוגדרו המשתנים של הספרייה tensorflow בצורה שהמודל יוכל לחשב את המודל של הרשת נויירונים בעזרת הכלים שהוגדרו ב- tensorflow. מייד לאחר מכן, אנו בנינו את המודל של הרשת נויירונים והגדרנו אותה. בבניית הרשת "שיחקנו" בכל פעם עם העומק של הרשת (כמות השכבות) ובכמות הנויירונים שבכל שכבה. יש לציין שהבחנו שכאשר אנו העמסנו כל שכבה בהרבה נויירונים (256 נויירונים) המודל לא הצליח ללמוד. לאחר בדיקות וניסויים הבנו שהמודל שלנו יהיה טוב אם יהיו לנו 4 hidden layers:

שכבה נסתרת ראשונה: 20 נויירונים

שכבה נסתרת שנייה: 40 נויירונים

שכבה נסתרת שלישית: 40 נויירונים

שכבה נסתרת רביעית: 4 נויירונים

וכמובן, שכבת הקלט מכילה נויירונים ככמות ה-features שלנו ושכבת הפלט מכילה נויירון אחד.

בכל ה-hidden layers השתמשנו בפונקציית אקטיבציה reLU ע"מ לפשט את חישוב הגרדיאנט עבור tensorflow ומכיוון שהיא מתאימה לבעיות קלאסיפיקציה כמו שיש לנו. כמו כן, דאגנו לבחור את פונקציית האקטיבציה עבור הנויירון שבשכבת הפלט להיות סיגמוייד.

בנינו את הרשת ע"י שימוש ב-tensorflow.layers.dense שמוסיפה שכבה חדשה לרשת שלנו (עם כמות הנויירונים שאנו מעוניינים ופונקציית אקטיבציה כפי שהחלטנו).

**חלק שלישי(model\_training):**

בחלק השלישי נלקחו הנתונים שנמצאים ב- train\_data\_x , train\_data\_y , ועל סמך הנתונים האלו המודל התחיל לחשב את פונקצית ה- logstic על מנת למצוא את המשקלים הטובים ביותר. בתוכנית נעשו 10000 חזרות על הנתונים עד שהמודל סיים את שלב האימון.

**חלק רביעי(model\_testing):**

בחלק הרביעי אחרי מציאת המשקלים בחלק השלישי יבדק עד כמה המודל מדויק על ידי שימוש במידע הנותר שלא נעשה בו שימוש לצורך למידה ב- test\_data\_x , test\_data\_y על ידי הרצה של דגימה והשוואה בין התוצאה של המודל לבין התוצאה האמיתית. במהלך ה-test אנו מחשבים את ה-Loss וה-Accuracy ובסופו אנו מציגים אותם.

**מסקנות:**

במהלך ה- train(2217 דוגמאות, עם 10000 חזרות) שנעשה על המודל דאגנו להדפיס את ה-Loss וה-Accuracy ע"מ שנוכל להבחין בתהליך הלמידה והתוצאות שהתקבלו הן:

start training the model

Step: 0 Loss: 0.688 Acc: 50.34%

Step: 500 Loss: 0.534 Acc: 78.98%

Step: 1000 Loss: 0.467 Acc: 86.42%

Step: 1500 Loss: 0.436 Acc: 90.26%

Step: 2000 Loss: 0.422 Acc: 91.88%

Step: 2500 Loss: 0.414 Acc: 91.97%

Step: 3000 Loss: 0.409 Acc: 92.87%

Step: 3500 Loss: 0.405 Acc: 93.19%

Step: 4000 Loss: 0.401 Acc: 93.69%

Step: 4500 Loss: 0.396 Acc: 94.14%

Step: 5000 Loss: 0.391 Acc: 95.04%

Step: 5500 Loss: 0.396 Acc: 94.18%

Step: 6000 Loss: 0.390 Acc: 95.04%

Step: 6500 Loss: 0.387 Acc: 95.40%

Step: 7000 Loss: 0.385 Acc: 95.62%

Step: 7500 Loss: 0.384 Acc: 95.81%

Step: 8000 Loss: 0.383 Acc: 95.53%

Step: 8500 Loss: 0.382 Acc: 95.85%

Step: 9000 Loss: 0.382 Acc: 95.99%

Step: 9500 Loss: 0.383 Acc: 96.12%

Step: 10000 Loss: 0.380 Acc: 96.08%

finish training the model

ניתן להבחין שטרם תחילת הלמידה של המודל אחוז הדיוק קרוב ל-50%, שמשמעותו בבעיית קלאסיפיקציה בינארית – ניחוש. לאחר סיום הלמידה המודל הגיע לאחוז דיוק גבוה מאוד: 96.08%.

לאחר שביצענו את שלב ה- test(עם 951 דוגמאות) על המודל קיבלנו את התוצאות הבאות:

Loss: 0.433 Accuracy: 93.27%

בהשוואה למודל ה-Logistic Regression שמימשנו במטלה הקודמת, קיים הפרש של למעלה מ-13% באחוז הדיוק לאחר ה-test. במודל Logistic Regression קיבלנו אחוז דיוק 80.7% ובמודל רשת הנויירונים שמימשנו במטלה זו קיבלנו אחוז דיוק 93.27%, שזה שיפור משמעותי.

יש לציין שערכנו מספר ניסויים בעיצוב במודל של הרשת נויירונים. במהלך הניסויים הבחנו שכאשר אנו מעמיסים יתר על המידה בכמות הנויירונים שבכל שכבה ביחד לכמות ה- featuresשלנו המודל לא הצליח ללמוד, וכאשר הוספנו להעמיק את הרשת זמן הלימוד התארך. בסוף הצלחנו למצוא את האיזון ולבנות מודל שיהיה מספיק חזק ללמידת הבעייה ולדעת לסווג קול למיגדר באחוז דיוק גבוה.