3. Linear Neural Networks

פרק זה עוסק בבעיות רגרסיה – כיצד ניתן בעזרת סט דוגמאות נתון לבנות מודל המסוגל לספק מידע על נקודות חדשות שיגיעו וחסר עליהן מידע. המודלים שיוצגו בפרק זה מתייחסים לדאטא שניתן למצוא עבורו הפרדה לינארית, כלומר, ניתן למצוא קווים ליניאריים המחלקים את הדאטא לקבוצות שונות. החלק הראשון של הפרק יעסוק ברגרסיה לינארית (Linear regression) והחלק השני יעסוק ברגרסיה לוגיסטית (Logistic regression). לבסוף יוצג מבנה שקול לבעיות הרגרסיה בעזרת רשת נוירונים פשוטה, ומבנה זה יהיה הבסיס לפרק הבא העוסק ברשתות נוירונים עמוקות, הבאות להתמודד עם דאטא שאינו ניתן לבצע עבורו הפרדה לינארית.

3.1 Linear Regression

3.1.1 The Basic Concept

המודל הפשוט ביותר הינו linear regression. מודל זה מנסה למצוא קשר לינארי בין מספר משתנים או מספר פיצ'רים. בהנחה שמתקיים יחס לינארי בין סט משתנים בלתי תלויים $\boldsymbol{x} \in \mathbb{R}^d$ לבין משתנה תלוי $y \in \mathbb{R}$, ניתן לכתוב את הקשר ביניהם בצורה הבאה:

$$\hat{y} = w^T x + b = w_1 x_1 + w_2 x_2 + \cdots w_d x_d + b$$

כאשר $\boldsymbol{w} \in \mathbb{R}^d$ הם המשקלים ו-$b \in \mathbb{R}$ נקרא bias.

דוגמא: ניתן לטעון כי מחיר הבתים באזור מסוים נמצא ביחס לינארי למספר פרמטרים: גודל הדירה, איזה קומה היא נמצאת, וכמה שנים הבניין קיים. תחת הנחה זו, יש לבחון את המודל עבור דוגמאות ידועות ובכך למצוא את המשקלים וה-$bias$. לאחר מכן ניתן יהיה לקחת את המודל ולנחש את מחיר הדירה עבור בתים שמחירם לא ידוע, אך הפרמטרים שלהם כן נתונים.

בכדי לבנות מודל המאפשר לשערך בצורה טובה את y בהינתן סט פיצ'רים, יש לדעת את המשקלים וה-bias. כיוון שהם לא ידועים, יש לחשב אותם בעזרת סט של דוגמאות ידועות. ראשית יש להגדיר פונקציה מחיר (Loss), הקובעת עד כמה הביצועים של מודל מסוים טובים. פונקציית המחיר היא פונקציה של הפרמטרים הנלמדים - $L(\boldsymbol{w}, b)$, והבאתה למינימום תספק את הערכים האופטימליים של המשקלים וה-bias. פונקציית מחיר מקובלת הינה השגיאה הריבועית הממוצעת (MSE) – המחשבת את ריבוע ההפרש בין החיזוי לבין הפלט האמיתי:

$$L^{(i)}(\boldsymbol{w}, b) = \frac{1}{2}(y_i - \hat{y}_i)^2$$

כאשר נתונות n דוגמאות ידועות, יש לסכום את כל ההפרשים האלו:

$$L(\boldsymbol{w}, b) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y_i - \hat{y}_i)^2 = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y_i - w^T x_i - b)^2$$

כעת בשביל למצוא את הפרמטרים האופטימליים, יש למצוא את $\mathbf{w}, \mathrm{b}$ שמביאים את פונקציית המחיר למינימום:

$$\hat{\boldsymbol{w}}, \hat{b} \equiv \hat{\theta} = \arg\min L(w, b)$$

עבור המקרה הסקלרי בו $d = 1$, כלומר יש פיצ'ר יחיד ומנסים למצוא קשר בינו לבין פלט מסוים, הקשר הלינארי הוא $\hat{y} = wx + b$. עבור המקרה הזה, פונקציית המחיר תהיה:

$$L(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y_i - w^T x_i - b)^2$$

ובכדי למצוא אופטימום יש לגזור ולהשוות ל-0:

$$\frac{\partial L}{\partial w} = \frac{1}{n}\sum_{i=1}^{n}(y_i - w^T x_i - b) \cdot (-x_i) = 0$$

$$\frac{\partial L}{\partial b} = \frac{1}{n}\sum_{i=1}^{n}(y_i - w^T x_i - b) \cdot (-1) = 0$$

מתקבלות סט משוואות לינאריות:

$$w\sum x_i^2 + b\sum x_i = \sum y_i x_i$$
$$w\sum x_i + bn = \sum y_i$$

ובכתיב מטריציוני:

$$\begin{pmatrix} \sum x_i^2 & \sum x_i \\ \sum x_i & n \end{pmatrix}\begin{pmatrix} w \\ b \end{pmatrix} = \begin{pmatrix} \sum y_i x_i \\ \sum y_i \end{pmatrix}$$

על ידי הצבה של הדוגמאות הנתונות ניתן לקבל את הפרמטרים של הקשר הלינארי.

לשם הנוחות ניתן לסמן את ה-bias כפרמטר נוסף:

$$\hat{y} = w^T x + b = (w^T\ b)\begin{pmatrix} x \\ 1 \end{pmatrix} = \widetilde{w}^T \tilde{x}, \qquad \widetilde{w}, \tilde{x} \in \mathbb{R}^{d+1}$$

עבור המקרה הוקטורי יש n דוגמאות, כלומר יש n פיצ'רים בלתי תלויים ומנסים למצוא את הקשר ביניהם לבין פלט מסוים. במקרה זה $X_{n\times d+1} = (x_1, \dots, x_n)^T, Y = (y_1, \dots, y_n)^T$, ופונקציית המחיר הינה:

$$L(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y_i - w^T x_i)^2$$

המינימום של הביטוי הזה שקול למינימום של $||Y - Xw||^2$:

$$\frac{\partial L}{\partial w} = \frac{1}{n}\sum_{i=1}^{n}(y_i - w^T x_i)\cdot(-x_t) = 0$$
$$\rightarrow X^T(Xw - Y) = 0$$
$$\widehat{w} = (X^T X)^{-1} X^T Y$$

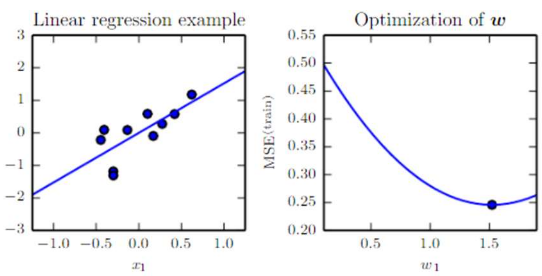
ובהינתן סט דוגמאות:

$$X = \begin{pmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix}, \quad \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

אזי הפתרון של הרגרסיה הלינארית הינו:

$$\widehat{w} = (X^T X)^{-1} X^T Y = \begin{pmatrix} \sum x_i^2 & \sum x_i \\ \sum x_i & n \end{pmatrix}^{-1}\begin{pmatrix} \sum y_i x_i \\ \sum y_i \end{pmatrix}$$

דוגמא למציאת קו הרגרסיה והמשקל האופטימלי עבור בעיה סקלרית:



איור 3.1 רגרסיה לינארית אופטימלית עבור סט דוגמאות נתון (שמאל) ואופטימיזציה עבור המשקל w ביחס לפונקציית המחיר (ימין).

3.1.2 Gradient Descent

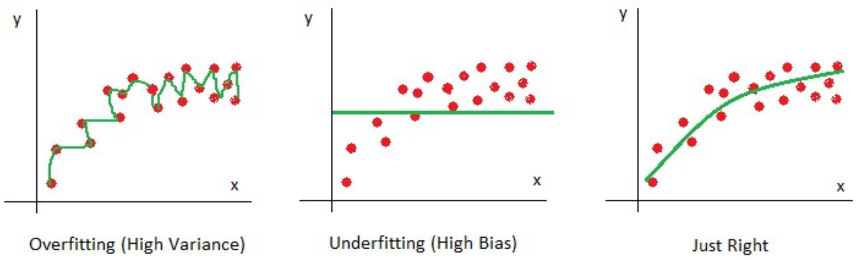
הרבה פעמים מציאת המינימום של פונקציית המחיר היא משימה קשה. דרך מקובלת להתמודד עם חישוב הפרמטר האופטימלי היא שיטת gradient descent (GD). בשיטה זו מתחילים מניחוש מסוים עבור הפרמטרים, וכל פעם מבצעים צעד לכיוון הגרדיאנט השלילי. הגרדיאנט הוא הנגזרת של הפונקציה, והוא מגדיר את הכיוון שערך הפונקציה עולה בו בצורה מקסימלית. אם לוקחים את הכיוון השלילי של הגרדיאנט, בעצם הולכים לכיוון בו יש את הירידה הכי גדולה, ולכן כדי להגיע למינימום יש לבצע את הצעד בכיוון הגרדיאנט השלילי. בכדי להימנע מהיקלעות לנקודת אוכף, מוסיפים איבר הנקרא learning rate (lr) (מסומן באות ϵ). מבצעים את הגזירה ושינוי הפרמטרים באופן איטרטיבי עד נקודת עצירה מסוימת. באופן פורמלי, עבור ניחוש התחלתי θ_0, בכל צעד יבוצע הקידום באופן הבא (העדכון מתבצע באופן סימולטני עבור כל θ_{j+1}):

$$\hat{\theta}_{j+1} = \hat{\theta}_j - \epsilon \cdot \frac{\partial}{\partial \theta_j} L(\hat{\theta}_j)$$

קידום זה יבוצע שוב ושוב עד התכנסות לערך מסוים. כיוון שהבעיה קמורה מובטח שתהיה התכנסות למינימום, אך היא יכולה להיות איטית עקב צעדי עדכון גדולים או קטנים מדי. פרמטר ה-learning rate, ϵ, קובע את קצב ההתכנסות, לכן רצוי לבחור פרמטר לא קטן מדי כדי לא להאט את ההתכנסות ולא גדול מדי כדי למנוע התכנסות.

3.1.3 Regularization and Cross Validation

אחד האתגרים המרכזיים של בעיית הרגרסיה (ושאר בעיות הלמידה) הוא לפתח מודל שיהיה מוצלח לא רק עבור סט הדוגמאות הידוע (-סט האימון), אלא שיהיה מספיק טוב גם עבור דוגמאות חדשות ולא מוכרות (-טסט סט). כל מודל יכול לסבול מהטיה לשני כיוונים – Overfitting ו-Underfitting. Overfitting הוא מצב בו יש רעש במודל הנלמד, הנובע מהערכת יתר הניתנת לכל נקודה בסט האימון, ומבניית מודל מסדר גבוה. במצב זה המודל מתאים רק לסט האימון, אך הוא לא מצליח להכליל גם נקודות חדשות. Underfitting הוא המצב ההפוך – מודל בעל שונות גבוהה שלא מצליח למצוא קו מגמה המכיל מספיק מידע על הדוגמאות הנתונות.



איור 3.2 Overfitting – נתינת משקל יתר לכל נקודה גורמת למצב בו המודל הוא מסדר גבוה ובעל רעש חזק (שמאל). Underfitting – מודל בעל שונות גבוהה שלא מייצג בצורה מספיק טובה את המידע (אמצע). מצב מאוזן – מודל בעל שגיאה מינימלית, המתאר בצורה טובה את המידע, ובנוסף נמנע משגיאת יתר עבור דוגמאות חדשות (ימין).

בכדי להימנע מהטיות אלו, יש לבצע Regularization – הוספת אילוץ המונע מהמודל להיות מוטה באופן הפוגע בתוצאות. לאחר הוספת האילוץ, פונקציית המחיר תהיה בצורה:

$$\text{Regularized Loss} = \text{Loss Function} + \text{Constraint}$$

יש מספר דרכים לבצע את ה-Regularization:

<u>Ridge Regression / L2 Regularization</u>

דרך אחת לבצע את ה-Regularization היא להוסיף איבר נוסף המתייחס לריבוע הפרמטרים:

$$L(\theta) = \text{MSE}_{\text{train}} + \lambda w^T w = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y_i - w^T x_i)^2 + \lambda \|w\|^2$$

כעת האופטימום של הביטוי הינו:

$$\hat{w} = (X^T X + \lambda I)^{-1} X^T Y$$

הוספת האילוץ גורמת לכך שבנוסף לחיזוי מדויק של משתנה המטרה, המודל מנסה למזער את ריבוע הפרמטרים, ובכך לנסות להקטין עד כמה שניתן את הערך של כל פרמטר ולהימנע ממצב בו נותנים משקל יתר לחלק מהפרמטרים. למעשה האילוץ מקטין את השונות של המודל ובכך עשוי למנוע overfitting.

Lasso / L1 Regularization

דרך נוספת לבצע את ה-Regularization היא להוסיף אילוץ המתייחס לערך המוחלט של הפרמטרים:

$$L(\theta) = \text{MSE}_{\text{train}} + \lambda|w| = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y_i - w^T x_i)^2 + \lambda|w|$$

הוספת האילוץ מכריחה את סכום הפרמטרים להיות כמה שיותר קטן, כדי למזער כמה שניתן את פונקציית המחיר. בפועל אילוץ זה מביא ל"רידוד משקלים", כלומר כופה חלק מהמקדמים להיות אפס, וכך למעשה יש מעין feature selection – בחירת הפרמטרים המשמעותיים יותר.

Elastic Net

ניתן לשלב בין Ridge Regression לבין Lasso, ובכך לנסות לכוון את המודל עבור היתרונות של כל שיטה – גם להימנע מנתינת משקל יתר לפרמטרים וגם ניסיון לאפס פרמטרים ולקבל מודל פשוט ככל הניתן. פונקציית המחיר במקרה זה תהיה מהצורה:

$$L(\theta) = \text{MSE}_{\text{train}} + \lambda\|w\|^2 + \lambda_2|w|$$

עבור כל אחת מהדרכים, יש למצוא את הפרמטר λ האופטימלי עבור ה-Regularization (במקרה של Elastic Net, הפרמטר λ הוא למעשה וקטור: $\vec{\lambda} = [\lambda_1, \lambda_2]$). שיטה מקובלת למציאת λ האופטימלי היא Cross validation – חלוקת n הדוגמאות של סט האימון ל-k קבוצות, אימון כל תתי הקבוצות מלבד אחת, ואז בדיקת הפרמטרים שהתקבלו בשלב האימון על הקבוצה שנותרה. בכל איטרציה מוציאים חלק מסט הדוגמאות והופכים אותן לטסט סט, וכך מוצאים את הפרמטר λ האופטימלי המונע מהמשקלים להגיע מ-fitting (בדרך כלל לוקחים את הממוצע של כל ה-λ מכל האיטרציות). נפוץ להשתמש ב-$k = 5$, ולמעשה עבור בחירה טיפוסית זו יהיו 5 איטרציות, שבכל אחת מהן האימון יתבצע על 80% מסט האימון, ולאחר מכן תתבצע הבדיקה של הפרמטרים שנלמדו על ה-20% הנותרים.

איור 3.3 Cross validation עם חלוקה ל-5 קבוצות ($k = 5$).

בחירה של $k = n$ נקראת leave-one out cross validation כיוון שלמעשה בכל איטרציה יש דוגמא אחת בלבד שלא נכללת בסט האימון ועליה מתבצעת הבדיקה של הפרמטרים שנלמדו.

3.1.4 Linear Regression as Classifier

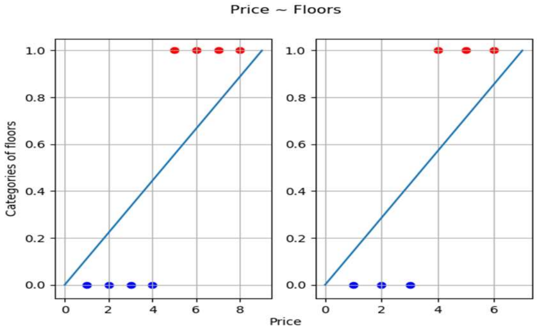
משימת סיווג מוגדרת באופן הבא: בהינתן סט פרמטרים מסוים $X = \{x_1, \ldots, x_n\}$ השייך לתצפית מסוימת, יש לסווג אותו לאחת מתוך m קטגוריות אפשריות: $y \in \{1, \ldots, m\}$. לדוגמא: נתונה תמונה בעלת n פיקסלים המייצגת חיה, ויש לקבוע איזו חיה היא, כאשר הבחירה נעשית מתוך הוקטור y. לרוב הוקטור y מורכב ממספרים שלמים, שכל אחד מהם מייצג בחירה מסוימת. בדוגמא של החיות, ניתן לקחת לדוגמא $m = 3$, כלומר $y \in \{1,2,3\}$, כאשר המספרים מייצגים את סט החיות $\{dog, cat, chicken\}$.

ניתן להשתמש במודל של רגרסיה לינארית למשימות של סיווג. עבור המקרה של $m = 2$, יש שתי קטגוריות אפשריות, ולמעשה יש צורך למפות כל נקודה לאחת משתי הקטגוריות. בעזרת רגרסיה לינארית ניתן לבצע מיפוי מ-

$\mathbb{R}$ ל-, $\{0,1\}$, כלומר כל נקודה במרחב ממופה לאחד משני ערכים אפשריים: קובעים ערך סף $T = 0.5$, ועבור נקודה חדשה x_n בודקים מה היחס בין הביטוי $w^T x_n + b$ לבין ערך הסף. אם הנקודה החדשה מקיימת: $w^T x_n + b < 0.5$ אז הנקודה החדשה תתויג בקטגוריה 1. אחרת, הנקודה החדשה תתויג בקטגוריה 0. באופן פורמלי:

$$y = sign(w^T x_{new} + b - 0.5) = \begin{cases} 1 & w^T x_{new} + b > 0.5 \\ 0 & w^T x_{new} + b < 0.5 \end{cases}$$

הבחירה בערך הסף $T = 0.5$ נובעת מכך שיש שתי קטגוריות $\{0,1\}$, וערך הסף נקבע להיות נקודת האמצע ביניהן. לדוגמא: נתונים n בתים ועבור כל אחד מהם ידוע מה מחירו והאם יש בו קומה אחת או שתיים. כעת רוצים לבחון את היחס בין המחיר למספר הקומות ולקבוע עבור מחיר בית נתון מה מספר הקומות שלו. במקרה זה יש 2 קטגוריות: $y \in \{0,1\} = \{1\ floor, 2\ floors\}$, ויש להיעזר בידע על n הבתים בכדי לבנות מודל מסווג. הדרך לעשות זאת היא לבצע רגרסיה לינארית, ואז כשבוחנים מחיר של בית, יש לבדוק אם $w^T \cdot price + b$ גדול מ-0.5 או קטן ממנו, כאשר (w, b) הם הפרמטרים של הרגרסיה הלינארית.



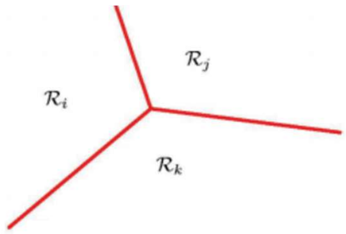
איור 3.4 רגרסיה לינארית כמסווג בינארי: מיפוי הנקודות בהתאם למיקומן ביחס לקו ההפרדה של הרגרסיה הלינארית. בדוגמא הימנית ערך הסף מתקבל עבור $x_T = 3.5$, כלומר $w^T \cdot 3.5 + b = 0.5$, ובדוגמא השמאלית ערך הסף מתקבל עבור $x_T = 4.5$. עבור כל בית חדש, בהינתן מחירו ניתן יהיה לסווג אותו לאחת משתי הקטגוריות, בהתאם ליחסו לערך הסף 0.5.

עבור n נקודות ידועות – $(x_1, y_1) \dots (x_n, y_n), y_i \in \{0,1\}$, פונקציית המחיר הינה:

$$L(\theta) = \sum_{i=1}^{n} 1_{\{y_i \neq sign(w^T x + b + 0.5)\}}$$

הפונקציה $L(\theta)$ מכילה סט של פרמטרים – $\theta = (\mathbf{w}, \mathrm{b})$. כיוון שהנגזרת של הפונקציה לפי כל אחד מהפרמטרים שלה w_i לא תלויה רק באותו פרמטר, קשה למצוא את θ המביאים למינימום את $L(\theta)$.

ניתן להרחיב את המסווג גם עבור מקרים בהם יש יותר משתי קטגוריות (multi-class). סט האימון יראה כמו במקרה הבינארי, ואילו y_i מכיל כעת m קטגוריות: $y_i \in \{1, \dots, m\}$. במקרים אלו יש לייצר מספר קווים לינאריים, המפרידים בין אזורים שונים. כדי לחשב את הקווים מבצעים את התהליך שנקרא one versus all, בו בכל פעם לוקחים קטגוריה אחת ובודקים מהו קו ההפרדה בינה לבין שאר הקטגוריות. הפרמטרים הנלמדים של קווי ההפרדה יהיו הסט המורכב מכל הפרמטרים של הרגרסיה: $\theta = \{w_1, b_1, \dots, w_m, b_m\}$.



איור 3.5 רגרסיה לינארית מרובה – הפרדה בין מספר אזורים שונים על ידי מספר קווים לינאריים.

במקרה הזה, נקודה חדשה תסווג לקטגוריה לפי הביטוי הבא:

$$y(x) = \arg\max_i (w_1^T x + b_1, \dots, w_m^T x + b_m)$$

וכל אזור יוגדר לפי:

$$R_i = \{x | y(x) = i\}$$

בדומה למקרה הבינארי, פונקציית המחיר תהיה:

$$L(\theta) = \sum_{i=1}^{n} 1_{\{y_i \neq \hat{y}_i\}} \; s.t \;\; \hat{y}_i = \arg\max_i (w_i^T x + b_i)$$

המסווג האופטימלי יהיה וקטור הפרמטרים המביא את פונקציית המחיר למינימום:

$$\hat{\theta} = \arg\min_{\theta} L(\theta)$$

גם במקרה הזה, כיוון שהנגזרת של פונקציית המחיר לפי כל פרמטר אינה תלויה רק באותו פרמטר, בפועל קשה למצוא את θ האופטימלי המביא את $L(\theta)$ למינימום.

3.2 Softmax Regression

3.2.1 Logistic Regression

המסווג הנוצר מהרגרסיה הלינארית הינו "מסווג קשה" – כל דוגמא חדשה שמתקבלת מסווגת לקטגוריה מסוימת, ואין שום מידע עד כמה הדוגמא הזו דומה לקטגוריות האחרות. מסווג כזה אינו מספיק טוב עבור מגוון בעיות, בהן מעוניינים לדעת לא רק את הקטגוריה, אלא גם מידע נוסף על היחס בין הדוגמא החדשה לבין כלל הקטגוריות. לדוגמא: בהינתן מימד של גידול מסוים רוצים לדעת אם הוא ממאיר או שפיר. במקרה זה ההכרעה היא לא תמיד חד משמעית, ויש עניין לדעת מה הסיכוי של הגידול להיות ממאיר או שפיר, שהרי יתכן והטיפול יהיה שונה בין מקרה בו יש 1% שהגידול הזה הוא מסוג מסוים לבין מקרה בו יש 40% שהגידול הוא מהסוג הזה. כדי להימנע מהסיווג הקטגורי, יש ליצור מודל הסתברותי, בו כל קטגוריה מקבלת הסתברות מסוימת. אחד המודלים הבסיסיים הינו רגרסיה לוגיסטית (Logistic regression). עבור המסווג הזה ראשית יש להגדיר את פונקציית הסיגמואיד:

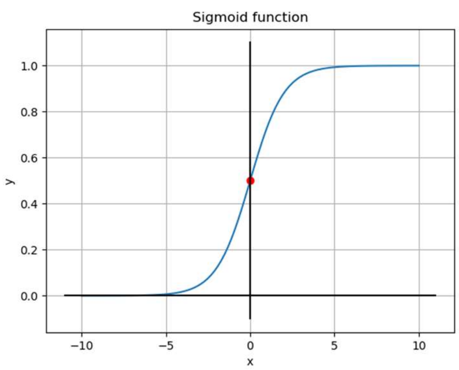
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

פונקציה זו רציפה על כל הישר, ובעזרתה ניתן להגדיר מסווג עבור המקרה הבינארי:

$$p(y = 1 | x; \theta) = \sigma(w^T x + b) = \frac{1}{1 + e^{-(w^T x + b)}}$$

$$p(y = 0 | x; \theta) = 1 - \sigma(w^T x + b) = 1 - \frac{1}{1 + e^{-(w^T x + b)}} = \frac{e^{-(w^T x + b)}}{1 + e^{-(w^T x + b)}}$$

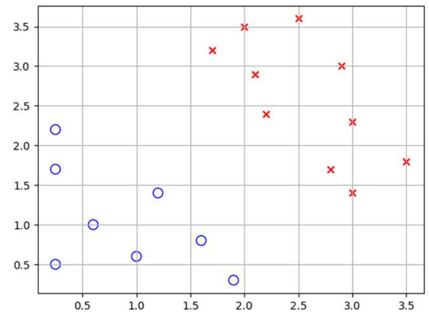
המסווג לוקח את קו ההחלטה הלינארי, ומעביר אותו בפונקציה המחזירה ערך בטווח $[0, 1]$, כאשר הערך המוחזר הוא ההסתברות להיות בקטגוריה מסוימת. בכדי להבין יותר טוב את משמעות המסווג, יש להסתכל על גרף הסיגמואיד:



איור 3.6 גרף הפונקציה: $y = \frac{1}{1+e^{-x}}$. הנקודה (0,0.5) מודגשת באדום.

כאשר הפונקציה $\theta(x) = w^T x + b$ שווה בדיוק ל-0, אזי $\sigma(w^T x + b) = 0.5$. המשמעות של התוצאה הזו היא שאם עבור סט פרמטרים x_n מתקיים $w^T x_n + b > 0$, אז ההסתברות של הנקודה הזו להיות משויכת לקטגוריה 1 גדולה מחצי, כיוון ש-$\sigma(w^T x + b) > 0.5$. באופן סימטרי אם $w^T x_n + b < 0$, אז ההסתברות של הנקודה x_n להיות

משויכת לקטגוריה 1 קטנה מחצי. כעת עולה השאלה מתי $w^T x + b = 0$, והתשובה היא שזה תלוי בקו ההפרדה שבין הקטגוריות. לשם המחשה נניח ונתונות מספר מדידות על שני פרמטרים - x_1, x_2, ועבור כל נקודה $(x_1 x_2)$ נתון גם מה הקטגוריה שלה $(y \in \{0,1\} = \{blue\ o, red\ x\})$:



איור 3.7 דוגמא למספר מדידות התלויות בשני פרמטרים x_1, x_2, ומשויכות לאחת משתי קטגוריות: $y \in \{0,1\} = \{blue\ o, red\ x\}$.

כיוון שנתונות הנקודות, ניתן לייצר בעזרתן קו רגרסיה. לצורך הדוגמא נניח שיש שלושה פרמטרים והם מקיימים: $[w_1, w_2, b]^T = [1, 1, -3]^T$. הפרמטרים האלו מרכיבים את הקו הלינארי $x_1 + x_2 = 3$, כלומר, עבור כל נקודה אם מתקיים $x_1 + x_2 > 3$ אזי היא תהיה מסווגת כ-'red x', אחרת היא תהיה מסווגת כ-'blue o'. קו זה הוא למעשה קו הפרדה שניתן בעזרתו לסווג נקודות חדשות. קו זה מקיים את המשוואה $w^T x + b = 0$, ולכן אם תהיה נקודה חדשה שגם מקיימת $w^T x_n + b = 0$, המשמעות היא שנקודה זו נמצאת בדיוק על קו ההפרדה. נקודה כזו תקבל הסתברות של 50% להיות משויכת לכל אחת מהקטגוריות. ככל שהנקודה החדשה תתרחק מקו ההפרדה, כך הביטוי $w^T x + b$ יתרחק מה-0, ולכן גם $\sigma(w^T x + b)$ יתרחק מהערך חצי ויתקרב לאחד מערכי הקצה 0 או 1, והמשמעות היא כמובן שיש יותר סיכוי שנקודה זה שייכת לקטגוריה אחת ולא לאחרת.

כמובן שניתן לקחת גם את המסווג ההסתברותי הזה ולהשתמש בו כמסווג קשה: עבור דוגמא חדשה לוקחים את ההסתברויות שלה לכל אחת מהקטגוריות, ומסווגים את הדוגמא לקטגוריה בעלת ההסתברות הגבוהה ביותר. במקרה הבינארי וקטור ההסתברויות הינו $\hat{y} = [p(y=1|x), p(y=0|x)]$, והקטגוריה של $\hat{y}$ תהיה $\arg\max_i \hat{y}_i$, שזהו ה-$\hat{y}_i$ בעל ההסתברות הגדולה ביותר.

3.2.2 Cross Entropy and Gradient descent

בכדי למצוא את הפרמטרים $\theta = (w, b)$ האופטימליים בהינתן n דוגמאות, ניתן להחליף את קריטריון השגיאה הריבועית הממוצעת בקריטריון אחר למזעור פונקציית המחיר – Cross entropy. קריטריון זה אומר שיש להביא למינימום את מינוס הלוג של סך הדוגמאות (הביטוי נובע משערוך הנראות המרבית – Maximum likelihood):

$$-\log P(Y|X;\theta) = -\frac{1}{n}\sum_{i=1}^{n} \log p(y_i|x_i;\theta) = L(\theta)$$

למעשה, יש למצוא את סט הפרמטרים $\hat{\theta}$ המביא את הביטוי למינימום: $\hat{\theta} = \arg\min_\theta L(\theta)$.

בכדי לחשב את הביטוי יש לפתח קודם את הביטוי עבור נגזרת הסיגמואיד:

$$\sigma(z) = \frac{1}{1+e^{-z}} \rightarrow \frac{\partial \sigma(z)}{\partial z} = \frac{-1}{(1+e^{-z})^2} \cdot e^{-z} \cdot (-1) = \frac{1}{1+e^{-z}} \cdot \frac{e^{-z}}{1+e^{-z}} = \sigma(z)\big(1-\sigma(z)\big)$$

כזכור, $p(y=0|x;w,b) = 1 - p(y=1|x;\theta) = 1 - \sigma(\theta)$, לכן יש לחשב גם את הנגזרת עבור $1 - \sigma(z)$:

$$\frac{\partial\big(1-\sigma(z)\big)}{\partial z} = -\sigma(z)\big(1-\sigma(z)\big)$$

בהתאם, הנגזרות של לוג סיגמואיד הן:

$$\frac{\partial \log \sigma(z)}{\partial z} = \frac{1}{\sigma(z)} \cdot \frac{\partial \sigma(z)}{\partial z} = \big(1-\sigma(z)\big)$$

$$\frac{\partial \log\big(1-\sigma(z)\big)}{\partial z} = \frac{1}{1-\sigma(z)} \cdot \frac{\partial\big(1-\sigma(z)\big)}{\partial z} = -\sigma(z)$$

כעת יש לשים לב שהנגזרת של $\log p(y=1|z)$ הינה $y-\sigma(z)=1-\sigma(z)$, והנגזרת של $\log p(y=0|z)$ הינה $y_i-\sigma(z)=0-\sigma(z)$. לכן אם $y\in\{0,1\}$, אז ניתן לרשום בקיצור: $\frac{\partial}{\partial\theta}\log p(y_i|z)=y_i-\sigma(z)$. במקרה של רגרסיה לוגיסטית, מחפשים את הנגזרת של $\frac{\partial}{\partial\theta}\log p(y_i|x_i;\theta)$, ולפי הפיתוח המקדים ניתן לרשום את זה כך:

$$\frac{\partial}{\partial\theta}\log p(y_i|x_i;\theta)=y_i-\sigma(w^Tx+b)\cdot\frac{\partial}{\partial x}(w^Tx+b)=y_i-p(y_i=1|x_i;\theta)\cdot x_i$$

כעת לאחר הפיתוח ניתן לחזור חזרה לביטוי $\arg\min_{\theta} L(\theta)$ ולהציב:

$$\frac{\partial L(\theta)}{\partial\theta}=-\frac{1}{n}\sum_{i=1}^{n}\frac{\partial}{\partial\theta}\log p(y_i|x_i;\theta)=-\frac{1}{n}\sum_{i=1}^{n}\big(y_i-\sigma(w^Tx+b)\big)x_i=-\frac{1}{n}\sum_{i=1}^{n}\big(y_i-p(y_i=1|\theta;x)\big)x_i$$

3.2.3 Optimization

בדומה לרגרסיה לינארית, גם כאן חישוב הערך האופטימלי של $\hat{\theta}$ יהיה איטרטיבי בשיטת gradient descent:

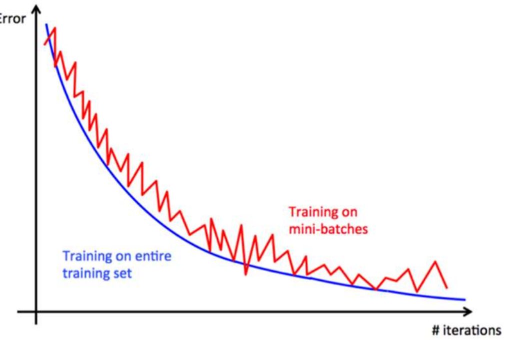
$$\hat{\theta}_{j+1}=\hat{\theta}_j-\epsilon\cdot\frac{\partial}{\partial\theta_j}L(\theta)$$

כאשר ϵ הוא הפרמטר של ה-learning rate. כיוון שפונקציית המחיר $L(\theta)$ קעורה, מובטח שתהיה התכנסות ל-$\hat{\theta}$.

במקרים רבים הדאטא סט הוא גדול, ולחשב את הגרדיאנט עבור כל הדאטא צורך הרבה חישוב. בכל צעד של קידום ניתן לחשב את הגרדיאנט עבור חלק מהדאטא, ולבצע את הקידום לפי הכיוון של הגרדיאנט המתקבל. למשל ניתן לבחור באופן אקראי נקודה אחת ולחשב עליה את הגרדיאנט. בחירה כזו נקראת Stochastic Gradient Descent (SGD), כיוון שבכל צעד יש בחירה אקראית של נקודה. חישוב בשיטת SGD יכול לגרום לשונות גדולה ככל שהחישוב מתקדם, ולכן עדיף לקחת מספר נקודות. חישוב הגרדיאנט בשיטה זו נקרא mini-batch learning (לעומת חישוב המתבצע על כל הדאטא הנקרא batch learning). באופן פורמלי, הגרדיאנט בשיטת mini-batch הינו:

$$\frac{\partial L}{\partial\theta}=\frac{\partial}{\partial\theta}\left[-\frac{1}{|V|}\sum_{i\in v}\log p(y_i|x_i;\theta)\right]\approx\frac{\partial}{\partial\theta}\left[-\frac{1}{n}\sum_{i=1}^{n}\log p(y_i|x_i;\theta)\right]$$

אמנם כל צעד הוא קירוב לגרדיאנט, אך החישוב מאוד מהיר ביחס לגרדיאנט המדויק, וזה יתרון משמעותי שיש לשיטה זו על פני batch learning. בנוסף, ניתן להוכיח שבשיטה זו מתקבל משערך חסר הטיה לגרדיאנט האמיתי.



איור 3.8 השגיאה של $\hat{\theta}$ כפונקציה של האיטרציות בשיטת gradient descent. הגרף הכחול מייצג את השגיאה בשיטת batch learning, בה הגרדיאנט בכל צעד מחושב על כל הדאטא, והגרף האדום מייצג את השגיאה בשיטת mini-batch learning, בה בכל צעד הגרדיאנט מחושב רק על חלק מהדאטא הנבחר באופן אקראי.

בדומה ל-linear regression, גם ב-logistic regression קיים עניין הרגולריזציה, שנועד למנוע מהמודל לתת משקל יתר לכל נקודה (Overfitting) או לא לייצג את הדאטא בצורה מספיק טובה (Underfitting). ניתן להוסיף למשל אילוץ ביחס לריבוע הפרמטר:

$$L(\theta)=-\frac{1}{n}\sum_{i=1}^{n}\log p(y_i|x_i;\theta)+\lambda\|\theta\|^2$$

ואז הנגזרת הינה:

$$\frac{\partial L}{\partial \theta} = -\frac{1}{n}\sum_{i=1}^{n}\frac{\partial}{\partial \theta}\log p(y_i|x_i;\theta) + 2\lambda\|\theta\|$$

הפרמטר λ האופטימלי מחושב על ידי ביצוע Cross validation על הדאטא.

3.2.4 SoftMax Regression – Multi Class Logistic Regression

בדומה ל-linear regression, גם ב-logistic regression ניתן להרחיב את המסווג גם עבור multi-class (מקרה בו יש יותר משתי קטגוריות). גם בהכללה למקרה מרובה קטגוריות יש מיפוי של כל קטגוריה להסתברות בתחום $[0,1]$ רק כעת הפונקציה בה משתמשים היא SoftMax במקום סיגמואיד. SoftMax היא פונקציה המופעלת על סדרה, והיא מוגדרת כך:

$$\text{SoftMax}(z_1,\dots,z_n) = \left(\frac{e^{z_1}}{\sum_{j=1}^{n} e_j^z},\dots,\frac{e^{z_n}}{\sum_{j=1}^{n} e_j^z}\right)$$

המונה מחשב אקספוננט בחזקת z_i, והמכנה מנרמל את התוצאה, כך שסך כל האיברים לאחר הפונקציה הוא 1. במקרה בו יש מספר קטגוריות – יש מספר קווי הפרדה, ולכל אחד מהם יש סט פרמטרים θ. בהינתן נקודה חדשה, ניתן בעזרת SoftMax לתת הסתברות לכל קטגוריה:

$$p(y=i|x;\theta) = \text{SoftMax}(w_1^T x + b,\dots,w_n^T x + b_n)$$

ואם מעוניינים לקבל סיווג קשה, לוקחים את האיבר בעל ההסתברות הגבוהה ביותר. גם במקרה זה פונקציית המחיר תהיה ה-Cross entropy:

$$L(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\log p(y_i|x_i;\theta)$$

נחשב את הנגזרת של הביטוי בתוך הסכום לפי θ_i:

$$\frac{\partial}{\partial \theta_i}\log p(y_i = s|x_i;\theta) = \frac{\partial}{\partial \theta_i}\log\frac{\exp(w_s^T x + b)}{\sum_{j=1}^{n}\exp\left(w_j^T x + b\right)} = \frac{\partial}{\partial \theta_i}\left(w_s^T x + b - \log\sum_{j=1}^{n}\exp\left(w_j^T x + b\right)\right)$$

$$= 1_{\{s=i\}}x - \frac{\exp\left(w_i^T x + b\right)x}{\sum_{j=1}^{n}\exp\left(w_j^T x + b\right)} = \left(1_{\{s=i\}} - p(y=i|x)\right)x$$

כאשר הסימון $1_{\{s=i\}}$ הינו 1 אם $s=i$ ו-0 אחרת. כעת ניתן להציב את הביטוי האחרון בנגזרת של $L(\theta)$:

$$\frac{\partial L}{\partial \theta_i} = -\frac{1}{n}\sum_{t=1}^{n}1_{\{y_t=k\}}x - p(y_t = i|x_t;\theta)$$

כעת ניתן לחשב את θ האופטימלי בשיטת gradient descent:

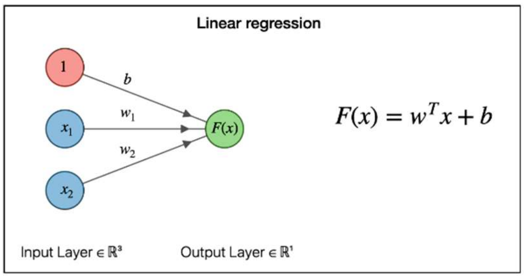
$$\theta_{i+1} = \theta_i - \epsilon\frac{\partial L}{\partial \theta}$$

3.2.5 SoftMax Regression as Neural Network

לשיטת logistic regression יש מספר יתרונות: היא יחסית קלה לאימון, מספקת דיוק טוב לדאטא-סטים פשוטים, יציבה ל-overfitting, מציעה סיווג הסתברותי ומתאימה גם למקרה בו יש יותר משתי קטגוריות. עם זאת, יש לה חסרון משמעותי – קווי ההפרדה של המודל הינם לינאריים, וזו הפרדה שאינה מספיק טובה עבור בעיות מורכבות. יש מגוון בעיות בהן על מנת לבנות מודל המסוגל להפריד בין קטגוריות שונות, יש צורך במנגנון הפרדה לא לינארי.

דרך מקובלת לבניית מודלים לא לינאריים היא שימוש ברשתות נוירונים עמוקות, ובכדי להבין את הקונספט שלהן היטב, ראשית יש לייצג את המודלים הלינאריים כשכבה של נוירונים, כאשר המודל הזה שקול לחלוטין לכל מה שהוצג עד כה. בעיית Linear regression לוקחת סט של פיצ'רים ומכפילה כל אחד מהם במשקל, ולאחר מכן סוכמת את

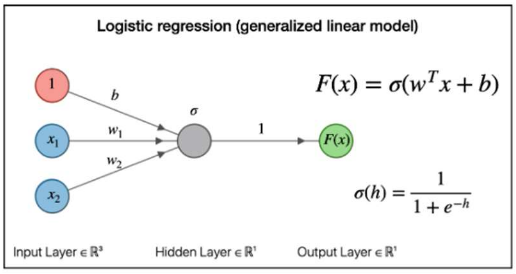
כל האלמנטים (בצירוף bias) לכדי משתנה יחיד הקובע מה הקטגוריה של סט זה. ניתן לייצג את המודל על ידי התיאור הגרפי הבא:



איור 3.9 ייצוג רגרסיה לינארית כרשת נוירונים עם שכבה אחת.

בתיאור זה יש 2 פיצ'רים המהווים את ה-input, וכל אחד מהם מחובר למוצא בתוספת הכפלה במשקל. בנוסף יש bias, ובצירוף הפיצ'רים המוכפלים במשקלים וה-bias מתקבל המוצא: $F(x) = w^T x + b = w_1 x_1 + w_2 x_2 + b$. כל עיגול באיור נקרא נוירון מלאכותי – אלמנט היכול לקבל קלט, לבצע פעולה חישובית ולהוציא קלט.

רגרסיה לוגיסטית ניתנת לתיאור באופן דומה, כאשר הנוירונים של סט ה-input לא מחוברים ישירות במוצא אלא עוברים דרך סיגמואיד במקרה הבינארי או דרך SoftMax במקרה בו יש יותר משתי קטגוריות:



איור 3.10 ייצוג רגרסיה לוגיסטית כרשת נוירונים עם שכבה אחת.

מלבד המעבר בפונקציית הסיגמואיד, יש הבדל נוסף בין הייצוג של הרגרסיה הלינארי לייצוג של הרגרסיה הלוגיסטית: בעוד הרגרסיה הלינארית מספקת במוצא מספר יחיד במוצא (מסווג קשה), הרגרסיה הלוגיסטית מספקת במוצא וקטור באורך של מספר הקטגוריות, באופן כזה שלכל קטגוריה יש הסתברות מסוימת שה-input שייך לאותה קטגוריה.

בפרק הבא יוצג מבנה בעל מספר שכבות של נוירונים, כאשר בין שכבה לשכבה יש פונקציה לא לינארית. באופן הזה המודל שיתקבל יהיה מיפוי של סט פיצ'רים באופן לא לינארי לוקטור הסתברויות במוצא. הגמישות של המודל תאפשר להתמודד עם משימות בעלות דאטא מורכב.