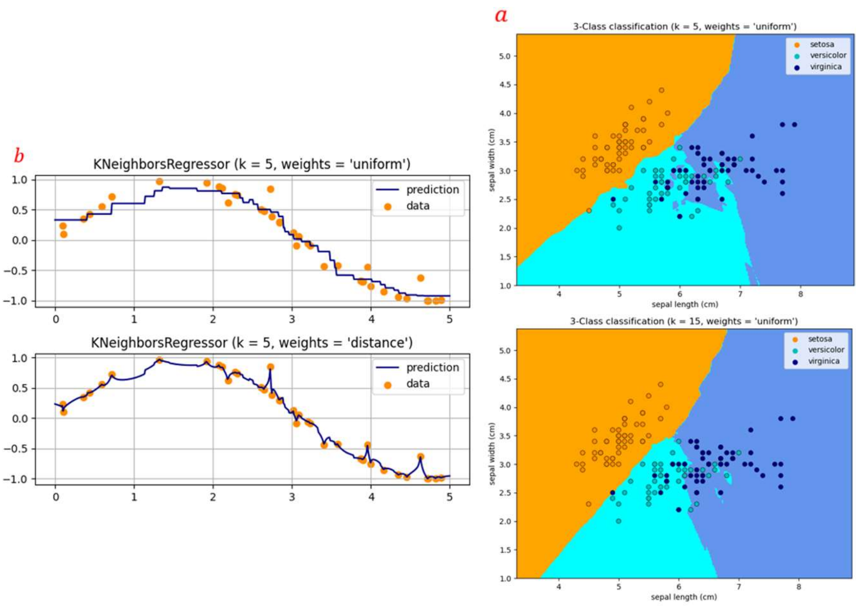
2. Machine Learning

2.1 Supervised Learning Algorithms

2.1.3 K-Nearest Neighbors (K-NN)

אלגוריתם השכן הקרוב הינו אלגוריתם של למידה מונחית, בו נתונות מספר דוגמאות ובנוסף ידוע ה-label של כל אחת מהן. אלגוריתם זה מתאים הן לבעיות סיווג (שיוך נקודה חדשה למחלקה מסוימת) והן לבעיות רגרסיה (נתינת ערך מאפיין לנקודה חדשה). האלגוריתם הינו מודל חסר פרמטרים, והוא מבצע סיווג לנתונים בעזרת הכרעת הרוב. עבור כל נקודה במדגם, המודל בוחן את ה-labels של K הנקודות הקרובות אליו ביותר, ומסווג את הנקודה לפי ה-label שקיבל את מרבית הקולות. מספר הנקודות הקרובות, K, הוא היפר-פרמטר שנקבע מראש.

אלגוריתם השכן הקרוב הוא אחד המודל הנפוצים והפשוטים ביותר בלמידת מכונה, וכאמור בנוסף לסיווג הוא מתאים גם לבעיות רגרסיה. המודל יפעל בצורה דומה בשני המקרים, כאשר ברגרסיה יתבצע שקלול של ממוצע בין השכנים הקרובים, ולא הכרעת הרוב, כלומר, התוצאה לא תהיה סיווג ל-label מסוים לפי הערך הנפוץ ביותר בקרב K השכנים הקרובים, אלא חישוב ממוצע של כל ה-labels השכנים. התוצאה המתקבלת היא ערך רציף, המייצג את הערכים בסביבת התצפית. ניתן להתחשב במרחק של כל שכן מהתצפית בצורה שווה (uniform), וניתן לתת משקל שונה לכל שכן בהתאם למרחק שלו מהנקודה אותה רוצים לחשב, כך שככל ששכן מסוים קרוב יותר לנקודה אותה רוצים לחשב כך הוא יותר ישפיע עליה, ביחס של הופכי המרחק בין השכן לבין הנקודה (distance).



איור 2.1 a) סיווג בעזרת אלגוריתם K-NN: מסווגים את המרחב לאזורים בהתאם ל-K השכנים הקרובים ביותר, כך שאם תבוא נקודה חדשה היא תהיה מסווגת בהתאם לצבע של האזור שלה, הנקבע כאמור לפי השכנים הקרובים ביותר. ניתן לראות שיש הבדל בין ערכי K שונים, וככל ש-K יותר גבוה ככה האזורים יותר חלקים ויש פחות מובלעות. b) רגרסיה בעזרת אלגוריתם K-NN: קביעת ערך ה-y בהתאם ל-K השכנים הקרובים ביותר. ניתן לתת משקלים שווים לכל השכנים, או לתת משקל ביחס למרחק של כל שכן מהנקודה אותה רוצים לחשב.

לעיתים נאמר על המודל שהוא "עצלן". הסיבה לכך היא שבשלב האימון לא מתבצע תהליך משמעותי, מלבד השמה של המשתנים וה-labels כאובייקטים של המחלקה, כלומר כל נקודה משויכת למחלקה מסוימת. עקב כך, כל מדגם האימון (או רובו) נדרש לצורך התחזית, מה שעשוי להפוך את המודל לאיטי כאשר יש הרבה דאטה. למרות זאת, המודל נחשב לאחד המודלים הקלאסיים הבולטים, בזכות היתרונות שלו. הוא פשוט וקל לפירוש, עובד היטב עם מספר רב של מחלקות, ומתאים לבעיות רגרסיה וסיווג. בנוסף הוא נחשב אמין במיוחד, כיוון שהוא לא מניח הנחות לגבי התפלגות הנתונים (כמו רגרסיה לינארית למשל).

מנגד, יש לו מספר חסרונות. עקב העובדה שהוא דורש את כל נתוני האימון בשביל התחזית, הוא עשוי להיות איטי כאשר מדובר על דאטה עשיר. מסיבה זו הוא גם אינו יעיל מבחינת זיכרון. מכיוון שהמודל דורש את כל נתוני האימון לצורך המבחן, כושר ההכללה שלו עשוי להיפגם (Generalization). ניקח לדוגמה מורה של כיתה בבית ספר, המנסה לסווג את התלמידים למספר קבוצות. אם יעשה זאת לפי צבע שיער ועיניים, לדוגמא, סביר להניח שלא יתקשה בכך;

אם לעומת זאת הוא ינסה לסווג לפי צבע שיער, עיניים, חולצה, מכנסיים, נעליים, וכו' – סביר שיתקל בקושי. במצב כזה, כל תלמיד רחוק מרעהו באופן שווה כיוון שאין שני תלמידים שזהים לחלוטין בכל הפרמטרים, מה שמקשה על חישוב המרחק. בעיה זו מכונה קללת הממדיות (Course of dimensionality), ולכן מומלץ להיעזר באמצעים להורדת הממד (Dimensionality reduction).

קושי נוסף הקיים במודל הוא הצורך בבחירת ה-K הנכון, מטלה שעשויה להיות לא קלה לעיתים. בכל מימוש של אלגוריתם השכן הקרוב, K הינו היפר-פרמטר שצריך להיקבע מראש. היפר פרמטר זה קובע את מספר הנקודות אשר האלגוריתם יתחשב בהן בעת בחירת סיווג התצפית. בחירת היפר-פרמטר קטן מידי, לדוגמא $K = 1$, יכולה לגרום למצב בו המודל מותאם יתר על המידה לנתוני האימון, מה שמוביל לדיוק גבוה בנתוני האימון, ודיוק נמוך בנתוני המבחן. מן העבר השני, כאשר K גבוה מידי, למשל $K = 100$, נוצר המצב ההפוך – מודל שמתחשב יותר מדי בדאטה ולא מצליח למצוא הכללה נכונה לסיווג. מומלץ לבחור K אי-זוגי בגלל אופן הפעולה של האלגוריתם – הכרעת הרוב. כאשר בוחרים K זוגי, עלולים להיתקל במצב של שוויון אשר עשוי להוביל לתוצאה מוטעית, ולכן כדי להימנע מתיקו כדאי לבחור K אי זוגי.

כמו אלגוריתמים רבים מבוססי מרחק, אלגוריתם השכן הקרוב רגיש לערכים קיצוניים (Outliers) ושימוש באלגוריתם ללא טיפול בערכים קיצוניים עשוי להוביל לתוצאות מוטות. מלבד זאת, חשוב לנרמל את הנתונים לפי שימוש במודל. הסיבה לכך היא שהאלגוריתם מבוסס מרחק; במצב זה, ייתכנו מרחקים בין תצפיות אשר עשויים להשפיע על החלטת המודל, למרות שמרחקים אלו הם חסרי משמעות לצורך הסיווג. דוגמא לכך היא משתנה שעושה שימוש ביחידות מידה שונות (מיילים\קילומטרים). ההחלטה האם להשתמש בקילומטרים או במיילים עלולה להטות את תוצאת המודל, למרות שבפועל לא השתנה דבר.

השיטה הנפוצה ביותר למדידת מרחק בין משתנים רציפים היא מרחק אוקלידי – עבור שתי נקודות במישור, המרחק ביניהם יחושב לפי הנוסחה: $d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. במידה ומדובר במשתנים בדידים, כגון טקסט, ניתן להשתמש במטריקות אחרות כגון מרחק המינג, המודד את מספר השינויים הדרושים בכדי להפוך מחרוזת אחת למחרוזת שנייה, ובכך למדוד את הדמיון ביניהן.

לפני שימוש באלגוריתם השכן הקרוב, יש הכרח לוודא שהמחלקות מאוזנות. במידה ומספר דוגמאות האימון באחת המחלקות גבוה מאשר בשאר המחלקות, האלגוריתם ייטה לסווג למחלקה זאת. הסיבה לכך היא שבשל מספרן הגדול, מחלקה זו צפויה להיות נפוצה הרבה יותר בקרב K השכנים של כל תצפית. הדבר עשוי להביא לתוצאות מוטות, ולכן יש לוודא מראש שאכן יש איזון בין המחלקות השונות.

2.2 Unsupervised Learning Algorithms

2.2.3 Mixture Models

אלגוריתם K-means מחלק n נקודות ל-K קבוצות על פי מרחק של כל נקודה ממרכז מסוים. בדומה ל-K-means גם אלגוריתם mixture model הוא אלגוריתם של clustering, אך במקום להסתכל על כל קבוצה של נקודות כשייכות למרכז מסוים, המודל משייך נקודות להתפלגויות שונות. המודל מניח שכל קבוצה היא למעשה דגימות של התפלגות מסוימת, וכל הדאטה הוא ערבוב דגימות ממספר התפלגויות. הקושי בשיטה זה הוא האתחול של כל קבוצה – כיצד ניתן לדעת על איזה דוגמאות לנסות ולמצוא התפלגות מסוימת? עקב בעיה זו, לעיתים משתמשים קודם באלגוריתם K-means על מנת לבצע חלוקה ראשונית לקבוצות, ולאחר מכן מנסים למצוא לכל קבוצה של נקודות התפלגות מסוימת.

ראשית נניח שיש k אשכולות, אזי נוכל לרשום את ההסתברות לכל אשכול:

$$p(y = i) = \alpha_i, i = 1, \dots k$$

וכמובן לפי חוק ההסתברות השלמה מתקיים $\sum_i \alpha_i = 1$.

בנוסף נניח שכל אשכול מתפלג נורמלית עם פרמטרים $\theta_i = (\mu_i, \sigma_i)$, אזי נקודה השייכת לאשכול i מקיימת:

$$x|y = i \sim \mathcal{N}(\mu_i, \sigma_i), i = 1 \dots k$$

אם מגיעה נקודה חדשה ורוצים לשייך אותה לאחד האשכולות, אז צריך למעשה למצוא את האשכול i שעבורו הביטוי $p(y = i|x)$ הוא הכי גדול. לפי חוק בייס מתקיים:

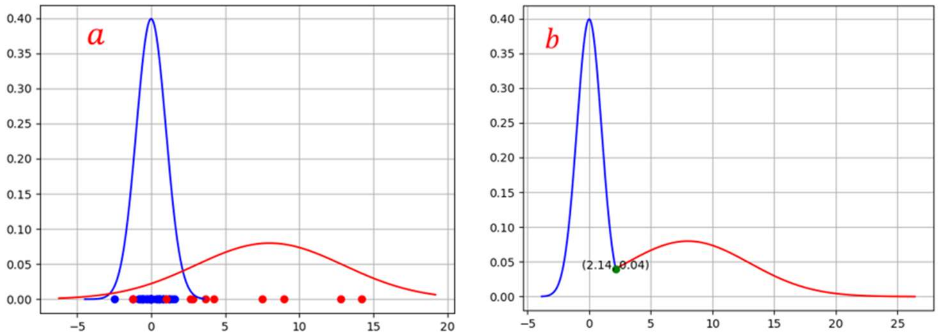
$$p(y = i|x) = \frac{p(y = i) \cdot p(x|y = i)}{p(x)}$$

המכנה למעשה נתון, כיוון שההתפלגות של כל אשכול ידועה ונותר לחשב את המכנה:

$$f(x)=f(x;\theta)=\sum_i p(y=i)f(x|y=i)=\sum_i \alpha_i \mathcal{N}(x;\mu_i,\sigma_i)$$

ובסך הכל:

$$p(y=i|x)=\frac{\alpha_i\cdot\mathcal{N}(x;\mu_i,\sigma_i)}{\sum_j \alpha_j \mathcal{N}\left(x;\mu_j,\sigma_j\right)}$$

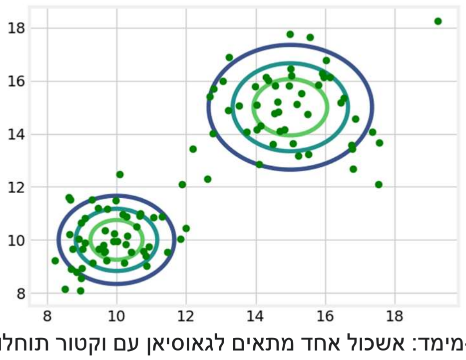


איור 2.2 a) תערובת של שני גאוסיאנים במימד אחד: בשלב ראשון מחלקים את הנקודות לשני אשכולות ומתאימים לכל אשכול התפלגות מסוימת. במקרה זה אשכול אחד (מסומן בכחול) הותאם להתפלגות $\mathcal{N}(0,1)$, ואשכול אחד (מסומן באדום) הותאם להתפלגות $\mathcal{N}(8,5)$.
b) נקודה חדשה x תסווג לאשכול הכחול אם $x<2.14$, כיוון שבתחום זה $\mathcal{N}(0,1)>\mathcal{N}(8,5)$. באופן דומה, הנקודה x תסווג לאשכול האדום אם $x>2.14$, כיוון שבתחום זה $\mathcal{N}(0,1)<\mathcal{N}(8,5)$.

כאמור, כדי לשייך נקודה חדשה x לאחד מהאשכולות, יש לבדוק את ערך ההתפלגות בנקודה החדשה. ההתפלגות שעבורה ההסתברות $p(x)$ היא הגדולה ביותר, היא זאת שאליה תהיה משויכת הנקודה. ההתפלגויות יכולות להיות בחד מימד, אך הן יכולות להיות גם במימד יותר גבוה. למשל אם מסתכלים על מישור, ניתן להתאים לכל אשכול התפלגות נורמלית דו-ממדית. במקרה ה-n מימדי, התפלגות נורמלית $X\sim\mathcal{N}(\mu,\Sigma)$ היא בעלת הצפיפות:

$$f_X(x_1,\dots,x_n)=\frac{1}{\sqrt{(2\pi)^n|\Sigma|}}e^{-\frac{1}{2}(X-\mu)^T\Sigma^{-1}(X-\mu)}$$

כאשר $|\Sigma|$ הוא הדטרמיננטה של מטריצת ה-covariance.



איור 2.3 תערובת של שני גאוסיאנים בדו-מימד: אשכול אחד מתאים לגאוסיאן עם וקטור תוחלות $\mu_1=[10,10]$ ומטריצת covariance: $\Sigma=\begin{bmatrix}1&0\\0&1\end{bmatrix}$, והאשכול השני מתאים לגאוסיאן עם וקטור תוחלות $\mu_1=[15,15]$ ומטריצת covariance: $\Sigma=\begin{bmatrix}2&0\\0&2\end{bmatrix}$.

כיוון שהאלגוריתם mixture model מספק התפלגויות, ניתן להשתמש בו כמודל גנרטיבי, כלומר מודל שיודע לייצר דוגמאות חדשות. לאחר התאמת התפלגות לכל אשכול, ניתן לדגום מההתפלגויות השונות ובכך לקבל דוגמאות חדשות.

2.2.4 Expectation–maximization (EM)

אלגוריתם מקסום התוחלת הינו שיטה איטרטיבית למציאת הפרמטרים האופטימליים של התפלגויות שונות, במקרים בהם אין נוסחה סגורה למציאת הפרמטרים. נתבונן על מקרה של Mixture of Gaussians, ונניח שיש אשכול מסוים המתפלג נורמלית עם תוחלת ושונות $\theta=(\mu,\sigma)$, ומשויכות אליו n נקודות. כדי לחשב את ההתפלגות של אשכול זה ניתן להשתמש בלוג הנראות המרבית:

$$L(\theta|x_1,\dots,x_n)=\log\prod_{i=1}^{n}\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}=\sum_{i=1}^{n}\log\frac{1}{\sqrt{2\pi\sigma^2}}-\frac{(x_i-\mu)^2}{2\sigma^2}$$

כדי למצוא את הפרמטרים האופטימליים ניתן לגזור ולהשוות ל-0:

$$\frac{\partial L(\theta)}{\partial\mu}=\sum_{i=1}^{n}\frac{x_i-\mu}{\sigma^2}\rightarrow\mu_{MLE}=\frac{1}{n}\sum_{i=1}^{n}x_i$$

$$\frac{\partial L(\theta)}{\partial\sigma^2}=\frac{1}{2\sigma^2}\left(-n+\frac{1}{\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2\right)\rightarrow\sigma_{MLE}^2=\frac{1}{n}\sum_{i=1}^{n}(x_i-\mu)^2$$

כעת נניח ויש k אשכולות וכל אחד מתפלג נורמלית. כעת סט הפרמטרים אותם צריך להעריך הינו:

$$\theta=\{\mu_1,\dots,\mu_k,\sigma_1^2,\dots,\sigma_k^2,\alpha_1,\dots,\alpha_k\}$$

עבור מקרה זה, הלוג של פונקציית הנראות המרבית יהיה:

$$L(\theta|x_1,\dots,x_n)=\log\prod_{i=1}^{n}\sum_{j=1}^{k}\alpha_j\mathcal{N}\left(x_i,\mu_j,\sigma_j^2\right)=\sum_{i=1}^{\mathrm{n}}\log\left(\sum_{j=1}^{k}\alpha_j\mathcal{N}\left(x_i,\mu_j,\sigma_j^2\right)\right)$$

אם נגזור ונשווה ל-0 נקבל בדומה למקרה הפשוט:

$$\sum_{i=1}^{n}\frac{1}{\sum_{j=1}^{k}\alpha_j\mathcal{N}\left(x_i,\mu_j,\sigma_j^2\right)}\alpha_j\mathcal{N}\left(x_i,\mu_j,\sigma_j^2\right)\frac{\left(x_i-\mu_j\right)}{\sigma_j^2}=0$$

נוסחה זו אינה ניתנת לפתרון אנליטי, ולכן יש הכרח למצוא דרך אחרת בכדי לחשב את הפרמטרים האופטימליים של ההתפלגויות הרצויות. נתבונן בחלק מהביטוי שקיבלנו:

$$\frac{1}{\sum_{j=1}^{k}\alpha_j\mathcal{N}\left(x_i,\mu_j,\sigma_j^2\right)}\alpha_j\mathcal{N}\left(x_i,\mu_j,\sigma_j^2\right)=\frac{p(y_i=j)\cdot p(x_i|y=j)}{p(x_i)}=p(y_i=j|x_i)\equiv w_{ij}$$

קיבלנו למעשה את הפוסטריור y_i (האשכול אליו רוצים לשייך את x_i), אך הוא לא נתון אלא הוא חבוי. כדי לחשב את המבוקש ננחש ערך התחלתי ל-θ ובעזרתו נחשב את y_i, ואז בהינתן y_i נבצע עדכון לפרמטרים – נבחן מהו סט הפרמטרים שמסביר בצורה הטובה ביותר את האשכולות שהתקבלו בחישוב ה-y_i. באופן פורמלי שני השלבים מנוסחים כך:

E-step – בהינתן סט נקודות x וערך עבור הפרמטר θ נחשב את האשכול המתאים לכל נקודה, כלומר כל נקודה x_i תותאם לאשכול מסוים y_i. עבור כל הנקודות y_i נחשב תוחלת ובעזרתה נגדיר את הפונקציה $Q(\theta,\theta_0)$, כאשר θ הוא פרמטר חדש ו-θ_0 הוא סט הפרמטרים הנוכחי:

$$Q(\theta,\theta_0)=\sum_{i=1}^{n}\sum_{j=1}^{k}p(y_i=j|x_i;\theta_0)\log p(y_i=j,x_i;\theta)=\sum_{i=1}^{n}\sum_{j=1}^{k}w_{ij}\log p(y_i=j,x_i;\theta)$$

$$\sum_{i=1}^{n}\mathbb{E}_{p(y_i|x_i;\theta_0)}\log p(y_i=j,x_i;\theta)$$

M-step – מחשבים את הפרמטר θ שיביא למקסימום את $Q(\theta,\theta_0)$ ואז מעדכנים את θ_0 ל-θ החדש:

$$\theta=\arg\max_{\theta}Q(\theta,\theta_0)$$

$$\theta_0\leftarrow\theta$$

חוזרים על התהליך באופן איטרטיבי עד להתכנסות.

עבור Mixture of Gaussians נוכל לחשב באופן מפורש את הביטויים:

$$Q(\theta, \theta_0) = \sum_{i=1}^{n} \sum_{j=1}^{k} w_{ij} \log p(y_i = j, x_i; \theta)$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{k} w_{ij} \log p(y_i = j; \theta) + \sum_{i=1}^{n} \sum_{j=1}^{k} w_{ij} \log p(x_i | y_i = j; \theta)$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{k} w_{ij} \log \alpha_j + \sum_{i=1}^{n} \sum_{j=1}^{k} w_{ij} \log \mathcal{N}\left(\mu_j, \sigma_j^2\right)$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{k} w_{ij} \log \alpha_j - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{k} w_{ij} \left(\log \sigma_j^2 + \frac{\left(x_i - \mu_j\right)^2}{\sigma_j^2} \right)$$

וכעת ניתן לגזור ולמצוא אופטימום:

$$\hat{\alpha}_j = \frac{1}{n} \sum_{i=1}^{n} w_{ij}$$

$$\hat{\mu}_j = \frac{\sum_{i=1}^{n} w_{ij} x_i}{\sum_{i=1}^{n} w_{ij}}$$

$$\hat{\sigma}_j^2 = \frac{\sum_{i=1}^{n} w_{ij} \left(x_i - \mu_j\right)^2}{\sum_{i=1}^{n} w_{ij}}$$

עבור התפלגויות שונות שאינן בהכרח נורמליות יש לחזור לביטוי של $Q(\theta, \theta_0)$ ולבצע עבורו את האלגוריתם.

נוכיח שהאלגוריתם משתפר בכל איטרציה, כלומר שעבור כל (θ, θ_0) מתקיים: $\log p(x; \theta) \geq \log p(x; \theta_0)$:

$$\log p(x; \theta) = \sum_{y} p(y|x; \theta_0) \log p(x; \theta) = \sum_{y} p(y|x; \theta_0) \frac{\log p(x, y; \theta)}{\log p(y|x; \theta)}$$

$$= \sum_{y} p(y|x; \theta_0) \left(\log p(x, y; \theta) - \log p(y|x; \theta)\right)$$

$$= \sum_{y} p(y|x; \theta_0) \log p(x, y; \theta) - p(y|x; \theta_0) \log p(y|x; \theta)$$

נשים לב שהאיבר הראשון הוא בדיוק $Q(\theta, \theta_0)$. האיבר השני לפי הגדרה הוא האנטרופיה של ההתפלגות $p(x|y; \theta_0)$:

$$H(\theta, \theta_0) = -\sum_{y} p(y|x; \theta_0) \log p(y|x; \theta_0)$$

כעת עבור שני ערכים שונים של θ מתקיים:

$$\log p(x; \theta) - \log p(x; \theta_0) = Q(\theta, \theta_0) + H(\theta, \theta_0) - Q(\theta_0, \theta_0) - H(\theta_0, \theta_0)$$

$$= Q(\theta, \theta_0) - Q(\theta_0, \theta_0) + H(\theta, \theta_0) - H(\theta_0, \theta_0)$$

לפי אי-שיוויון גיבס מתקיים $H(\theta, \theta_0) \geq H(\theta_0, \theta_0)$, לכן:

$$\log p(x; \theta) - \log p(x; \theta_0) \geq Q(\theta, \theta_0) - Q(\theta_0, \theta_0)$$

ולכן עבור כל עדכון של θ שמביא לאופטימום את $Q(\theta,\theta_0)$, הביטוי $Q(\theta,\theta_0)-Q(\theta_0,\theta_0)$ יהיה חיובי וממילא יהיה שיפור ב-$\log p(x;\theta)$.

2.2.5 Local Outlier Factor

אלגוריתם Local Outlier Factor הינו אלגוריתם של למידה לא מונחית למציאת נקודות חריגות (Outliers). האלגוריתם מחשב לכל נקודה ערך הנקרא Local Outlier Factor (LOF), ועל פי ערך זה ניתן לקבוע עד כמה הנקודה היא חלק מקבוצה או לחילופין חריגה ויוצאת דופן.

בשלב ראשון בוחרים ערך k מסוים. עבור כל נקודה x_i, נסמן את k השכנים הקרובים ביותר שלה ב-$N_k(x_i)$. כעת נגדיר את k-distance של כל נקודה כמרחק שלה מהשכן הרחוק ביותר מבין השכנים ב-$N_k(x_i)$. אם למשל $k=3$, אזי $N_k(x_i)$ הוא סט המכיל את שלושת השכנים הקרובים ביותר ל-x_i, וה-k-distance שלה הוא המרחק מהשכן השלישי הכי קרוב. חישוב המרחק בין שני שכנים נתון לבחירה – זה יכול להיות למשל מרחק אוקלידי, מרחק מנהטן ועוד. עבור בחירה של מרחק אוקלידי, ניתן להסתכל על k-distance כמעגל –הרדיוס של מעגל המינימלי המכיל את כל הנקודות השייכות ל-$N_k(x_i)$ הוא ה-k-distance.

לאחר חישוב ה-k-distance של כל נקודה, מחשבים לכל נקודה Local Reachability Density (LRD) באופן הבא:

$$LRD_k(x_i)=\frac{1}{\sum_{x_j\in N_k(x_i)}\frac{RD(x_i,x_j)}{k}}$$

כאשר $RD(x_i,x_j)=\max\left(\text{k}-\text{distance of }x_i, distance(x_i,x_j)\right)$. הגודל LRD מחשב את ההופכי של ממוצע המרחקים בין x_i לבין k השכנים הקרובים אליו. ככל שנקודה יותר קרובה ל-k-השכנים שלה כך ה-LRD שלה גדול יותר, ו-LRD קטן משמעותו שהנקודה יחסית רחוקה מאשכול הקרוב אליה.

בשלב האחרון בוחנים עבור כל נקודה x_i את היחס בין ה-LRD שלה וה-LRD של $N_k(x_i)$. היחס הזה הוא ה-LOF, והוא מחושב באופן הבא:

$$LOF_k(x_i)=\frac{\sum_{x_j\in N_k(x_i)}LRD(x_j)}{k}\times\frac{1}{LRD(x_i)}$$

הביטוי הראשון במכפלה הוא ממוצע ה-LRD של k השכנים של נקודה x_i, ולאחר חישוב הממוצע מחלקים אותו ב-LRD של הנקודה x_i עצמה. אם הערכים קרובים, אז ה-LOF יהיה שווה בקירוב ל-1, ואם הנקודה x_i באמת לא שייכת לאשכול של נקודות, אז ה-LRD שלה יהיה נמוך משמעותית מהממוצע של ה-LRD של השכנים שלה, וממילא ה-LOF שלה יהיה גבוה. אם עבור נקודה x_i מתקבל $LOF\approx 1$, אז סביר שהיא חלק מאשכול מסוים.

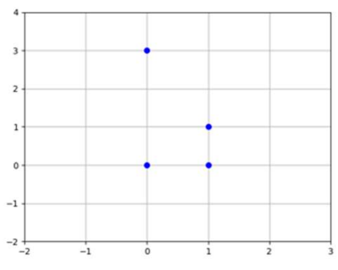
כדי להמחיש את התהליך נסתכל על הסט הבא: $\{A=(0,0),B=(1,0),C=(1,1),\ D=(0,3)\}$, ונקבע $k=2$. נחשב את ה-k-distance של כל נקודה במונחים של מרחק מנהטן:

$$k(A)=distance(A,C)=2$$

$$k(B)=distance(B,A)=1$$

$$k(C)=distance(C,A)=2$$

$$k(D)=distance(D,C)=3$$



נחשב את ה-LRD:

$$LRD_2(A)=\frac{1}{\frac{RD(A,B)+RD(A,C)}{k}}=\frac{2}{1+2}=0.667$$

$$LRD_2(B)=\frac{1}{\frac{RD(B,A)+RD(B,C)}{k}}=\frac{2}{2+2}=0.5$$

$$LRD_2(C) = \frac{1}{\frac{RD(C,B) + RD(C,A)}{k}} = \frac{2}{1+2} = 0.667$$

$$LRD_2(A) = \frac{1}{\frac{RD(D,A) + RD(D,C)}{k}} = \frac{2}{3+3} = 0.334$$

ולבסוף נחשב את ה-LOF:

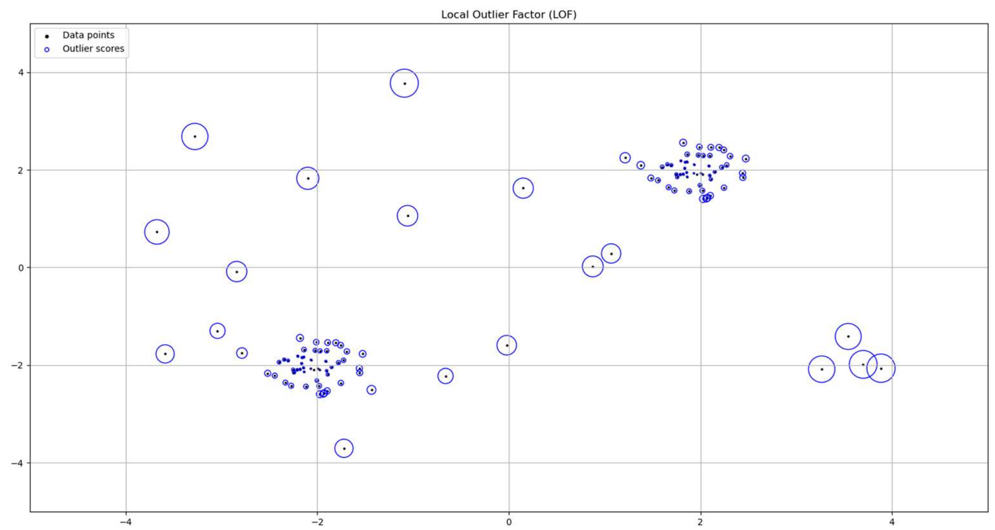
$$LOF_2(A) = \frac{LRD_2(B) + LRD_2(C)}{k} \times \frac{1}{LRD_2(A)} = 0.87$$

$$LOF_2(B) = \frac{LRD_2(A) + LRD_2(C)}{k} \times \frac{1}{LRD_2(B)} = 1.334$$

$$LOF_2(C) = \frac{LRD_2(B) + LRD_2(A)}{k} \times \frac{1}{LRD_2(C)} = 0.87$$

$$LOF_2(D) = \frac{LRD_2(A) + LRD_2(C)}{k} \times \frac{1}{LRD_2(D)} = 2$$

כיוון ש-$LOF_2(D) \gg 1$ באופן יחסי לשאר הנקודות, נסיק כי נקודה D היא outlier.



איור 2.4 Local Outlier Factor (LOF) – מציאת נקודות חריגות על ידי השוואת ערך ה-LRD של כל נקודה לממוצע ה-LRD של k השכנים שלה. ככל שה-LOF גדול יותר (העיגול הכחול), ככה הנקודה יותר רחוקה מאשכול של נקודות.

יש שני אתגרים מרכזיים בשימוש באלגוריתם זה – ראשית יש לבחור k מתאים, כאשר k יחסית קטן יהיה טוב עבור נקודות רועשות, אך יכול להיות בעייתי במקרים בהם יש הרבה מאוד נקודות הצמודות אחת לשנייה, ונקודה שמעט רחוקה מאוסף תזוהה כחריגה למרות שהיא באמת כן שייכת אליו. k גדול לעומת זאת יתגבר על בעיה זו, אך הוא לא יזהה נקודות חריגות שנמצאות בקירוב לאשכולות של נקודות. מלבד אתגר זה, יש צורך לתת פרשנות לתוצאות המתקבלות, ולהחליט על סף מסוים של LOF, שהחל ממנו נקודה מסווגת כחריגה. LOF קטן מ-1 הוא בוודאי לא outliner אך עבור ערכי LOF גדולים מ-1 אין כלל חד משמעי עבור איזה ערך הנקודה היא outliner ועבור איזה ערך היא לא. כדי להתמודד עם אתגרים אלו הוצעו הרחבות לשיטה המקורית, כמו למשל שימוש בסטטיסטיקות שונות המורידות את התלות בבחירת הערך k (LoOP – Local Outlier Probability), או שיטות סטטיסטיות העוזרות לתת פרשנות לערכים המתקבלים (Interpreting and Unifying Outlier Scores).