

מבוא לבינה מלאכותית – תרגול 8

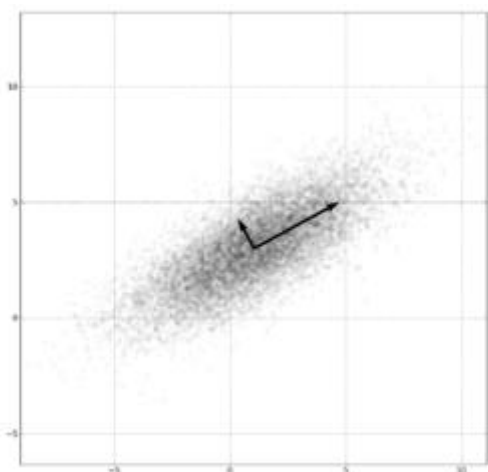
נושאים:

- PCA -
 - ICA -
 - Auto-Encoder -
 - MDS -
-

רקע – הורדת ממדים:

ככל שיש יותר פרמטרים להעריך במודל, כך המשימה לדייק בהערכה הזו דורשת יותר – חישוב מסובך או ארוך יותר, יותר דאטא ועוד. זו בעיה שלפעמים לא ניתן לפתור (במקרים רבים אין דאטא נוסף שאפשר להכניס), וזה עלול לגרום להתאמת יתר לנתונים הקיימים ופגיעה ביכולת ההכללה. כדי להתמודד עם הבעיה הזו, ישנם אלגוריתמים שמטרתם היא להקטין את מספר המשתנים והפרמטרים, במטרה מצד אחד להקל על התכנסות והערכה טובה ומצד שני לאבד כמה שפחות אינפורמציה מהדאטא המקורי.

:Principal Component Analysis (PCA)



זוהי שיטה שמניחה על הדאטא את הדבר הבא:

ישנה הצגה של הדאטא (אליה נשאף להגיע) ע"י וקטורים הלקוחים מהתפלגות נורמלית (אולי רב ממדית), עם תוחלת 0 ומטריצת Covariance אלכסונית. במקרה כזה, פיצ'רים בלתי מתואמים הם בהכרח בלתי תלויים.

נסתכל על 3 גישות להסתכלות על מה שמתבצע

באלגוריתם PCA:

1. שונות מקסימלית – ככל שיש יותר שונות בדאטא, כך נוכל להבחין בדברים, להפריד לאשכולות וכד'. מהסתכלות הפוכה, כאשר אין שונות, הדאטא מרוכז במקום אחד ולא מפוזר, כך שלא ניתן להבחין כמו שצריך במה שרוצים. אלגוריתם PCA ניתן לניסוח כבעיה שממקמת את השונות הנותרת לאחר הטרנספורמציה.

2. שגיאת השחזור מינימלית – נניח שכל וקטור במרחב הישן x_i מקורב על ידי וקטור במרחב החדש: $\hat{x}_i = \sum_{j=1}^k a_{ij} v_j$, ככה שלמעשה הייצוג החדש הוא וקטור המקדמים a_{ij} והווקטורים v_j לא משתנים בין וקטור לוקטור במרחב. אוסף וקטורי הפיצורים המקורבים \hat{x}_i (שיצג כמטריצה ויושווה למטריצה מתאימה של x_i) יהיה הקרוב ביותר לוקטור המקורי, מבין כל קירובי הווקטורים שניתן לעשות על ידי וקטורים $\{v_i\}_{i=1}^k$ כאלה.
3. הסרת קורלציות – חלק מהפיצורים יכולים להיות מתואמים אלה עם אלה, ולכן במרחב המקורי ישנם פיצורים מיותרים או מעורבבים אלה עם אלה. המטרה המקורית של PCA (היסטורית) הייתה לבצע טרנספורמציה למרחב הפיצורים כך שהפיצורים החדשים לא יהיו מתואמים (ובמקרה הגאומטרי – גם בלתי תלויים) אלה עם אלה.

כל הבעיות האלה נפתרות בעזרת אלגוריתם אחד, PCA:

אלגוריתם:

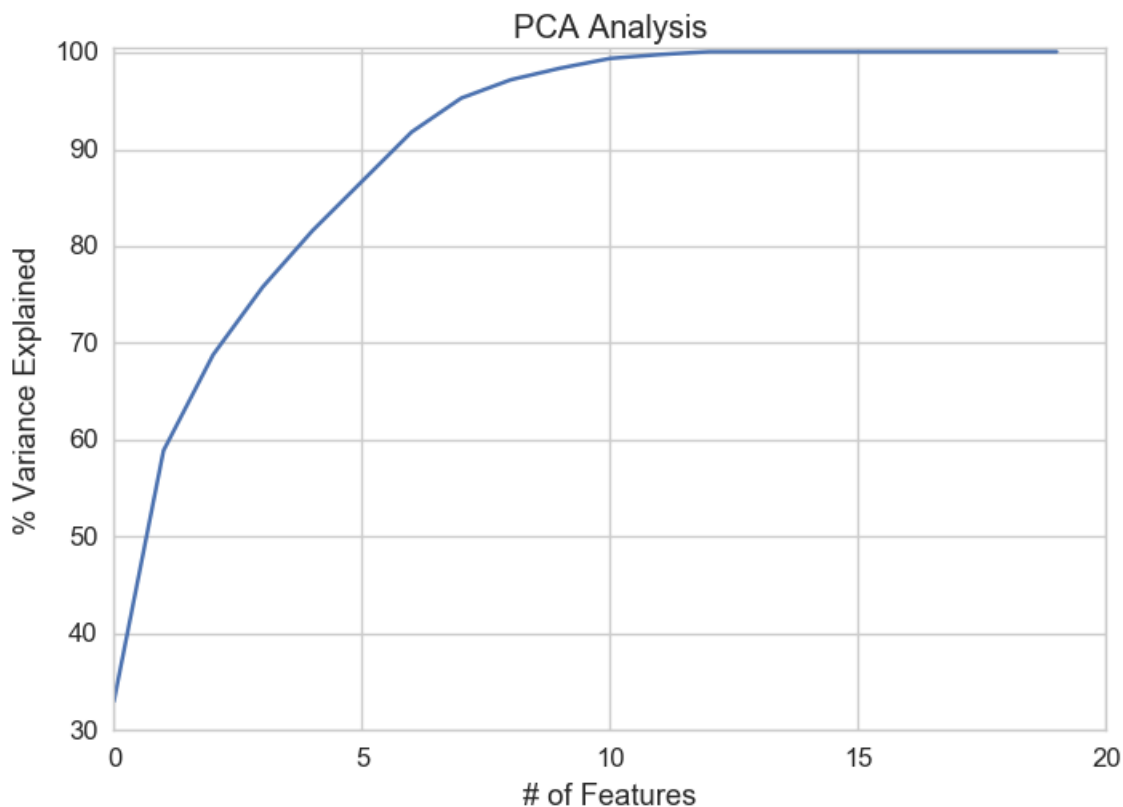
בהינתן אוסף נקודות $\{x_i\} \subseteq X$ כאשר X מרחב הפיצורים, מתבצעים:

1. "הלבנה" של הדאטא – חיסור הממוצע מכל נקודה, כדי שהממוצע של כל הפיצורים יהיה 0.
2. בניית מטריצת Covariance: $C = \sum_i \vec{x}_i \cdot \vec{x}_i^T$. שימו לב כי $\vec{x}_i \cdot \vec{x}_i^T$ היא מטריצה (וקטור \vec{x}_i) עמודה) חיובית, וכך גם C .
3. חישוב ערכים עצמיים גדולים ביותר ווקטורים עצמיים מתאימים להם (לגבי בחירת כמות הווקטורים, ראה בהמשך).
4. הפיצורים החדשים יהיו ההטלה של הפיצורים הישנים על מרחב הווקטורים העצמיים (אותם a_{ij} מהמשוואות לעיל).

נשים לב: הערכים העצמיים של מטריצת ה-covariance מעידים למעשה על כמה שונות מוסברת באמצעות הווקטור העצמי המתאים. ככל שהערך העצמי גדול יותר, ככה בעצם בכיוון הווקטור העצמי המתאים יש יותר שונות.

מה מקבלים ואיך בוחרים את הממד החדש:

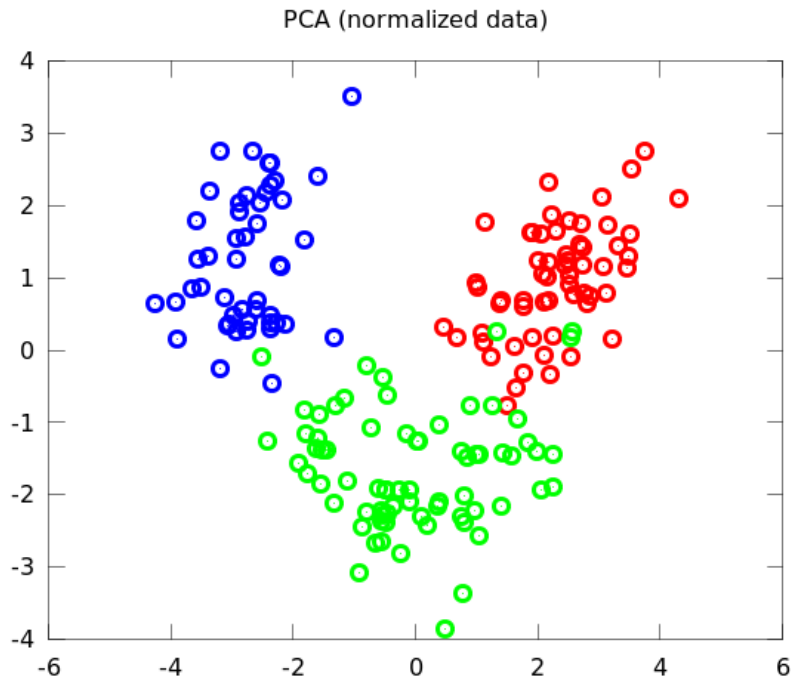
גרף טיפוסי של השונות המוסברת כתלות בערך העצמי הקטן ביותר שניקח:



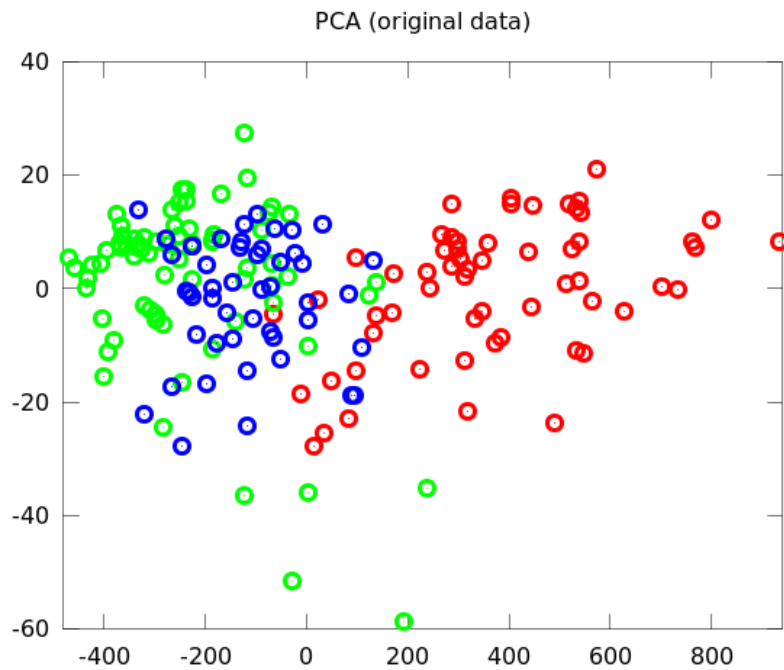
מכאן רואים שמספיק להעתיק את הדאטא למרחב בממד לא גבוה מאוד כדי לקבל שונות מספיקה. נשים לב: מצד אחד, יותר ממדים יסבירו יותר מהשונות הכוללת, ומצד שני, בעיות שנרצה לפתור תהיינה פשוטות יותר ככל שהממד יהיה נמוך יותר. אין כלל פורמלי לגבי כמה ממדים לבחור, וכנראה כדאי לבחור את הממד אליו מטילים את הדאטא על סמך התעסקות עם הדאטא יותר מאשר על סמך כללי אצבע.

דוגמה:

דאטאסט של יינות מ-3 סוגים שונים (נשתמש בתיוג אך ורק כדי לצייר), עם 13 פיצ'רים שונים. מנרמלים את הפיצ'רים כאן באמצעות z-score (גורמים לכלל אוסף ערכים שמתאימים לפיצ'ר אחד להיות בעלי שונות 1 ותוחלת 0), ואז מפעילים על התוצאה PCA. כאשר מציירים את הנקודות לפי שני הרכיבים הראשונים של כל אחת מהן לאחר הפירוק, מקבלים:



אגב, כאשר לא מנרמלים, מה שמתקבל משני הרכיבים בעלי השונות הגבוהה ביותר זה:



והמסקנה מכאן היא שלפעמים, כאשר יש הבדלים גדולים מאוד בסדרי הגודל של פיצ'רים, חשוב מאוד לנרמל.

הערה – באלגוריתמים שמיישמים PCA, לעתים משתמשים בפירוק לערכים סינגולריים (SVD) של מטריצת הדגימות במקום במטריצת ה-Covariance. בשיטה זו ניתן לחשב בדרך אחרת את הווקטורים העצמיים העיקריים. לרוב שיטה זו מועדפת על פני לכסון.

הערה – קיימת גרסת PCA עם kernel, לא ניכנס למצב זה מכיוון שהוא דומה ברעיונו ל-kernel SVM.

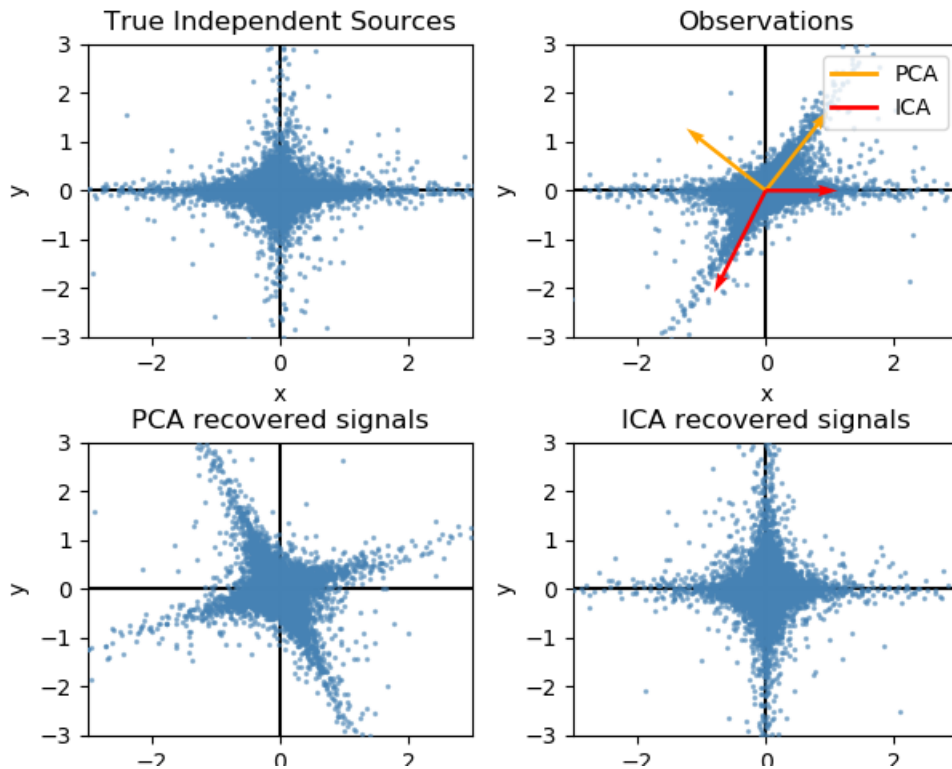
Independent Component Analysis (ICA)

זוהי שיטה אחרת להורדת ממדים, בהנחות שונות על המודל: במרחב החדש הדאטא יהיה מיוצג באמצעות וקטורים שלקוחים מהתפלגות לא נורמלית, עם תוחלת 0 וב"ת אלה באלה.

שתי הסתכלויות על הפירוק (מן הסתם אם נכנסים לפרטים יש בהן הבדלים קטנים, אולם הרעיון באופן אידיאלי [ולא ניתן למימוש כאלגוריתם] נותן תוצאות זהות):

1. אם כותבים $\vec{x} = A\vec{y} + \vec{\epsilon}$ כאשר \vec{x} הדאטא המקורי, \vec{y} ייצוג הדאטא במרחב החדש, A ההעתקה ו- $\vec{\epsilon}$ רעש, שחזור שגוי של \vec{y} מ- \vec{x} (שיוביל לערבוב של הרכיבים הנכונים והלא-נורמליים) ייתן בסיכוי גבוה וקטורים שלקוחים מהתפלגות נורמלית (לפי משפט הגבול המרכזי).
לכן, אלגוריתם ICA מנסה לשחזר את \vec{y} בצורה כזו שההתפלגות תהיה רחוקה ככל האפשר מהתפלגות נורמלית.
2. בעזרת תורת האינפורמציה, ההתפלגות במערכת בעלת מומנט שני (כלומר עם שונות סופית) שעבורה האנטרופיה מקסימלית היא התפלגות נורמלית.
לכן, ICA מנסה למזער את האנטרופיה.

דוגמה למצב בו הדאטא לא נורמלי ולכן המרחב המועתק באמצעות PCA לא מוצלח כמו המרחב המועתק באמצעות ICA:



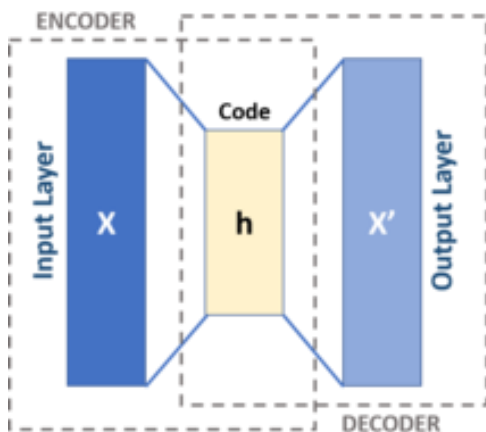
:Auto-Encoder

בדרך זו משתמשים ברשתות נוירונים על מנת להוריד ממדים. מודל כזה מורכב משני חלקים: Encoder ו-Decoder.

ה-Encoder מקבל את הקלט המקורי x ומחזיר ייצוג שלו בפחות ממדים, $z = enc(x)$.

ה-Decoder מקבל את הקלט המכווץ ומטרתו לשחזר ממנו את הקלט x כמה שיותר טוב:

$\hat{x} = dec(z)$ כך ש- $d(x, \hat{x})$ יהיה קטן ככל האפשר (d היא מטריקה כלשהי, למשל מרחק אוקלידי רגיל).



נשים לב שזהו אכן תהליך למידה לא מפוקח – לוקחים קלט x ומנסים לשחזר אותו. אין לנו תיוגים, אבל המודל מסוגל ללמוד כך כי יש לנו פונקציית $loss$.

לאחר שהמודל למד היטב את הדאטא (יש מודלים שמכניסים גם רגולריזציה ודברים מסובכים ומתקדמים יותר שנחקרים ונעשה בהם שימוש כיום, אבל הם מורכבים מדי עבורנו), הייצוג z הוא הווקטור החדש שבו משתמשים לכל מה שנרצה לעשות.

כמובן שאם z יהיה בממד נמוך מדי הלמידה לא תהיה טובה מספיק, ומצד שני אם z יהיה בממד גבוה מדי אז כנראה נוכל ללמוד אבל התוצאה תהיה פחות יעילה לשימוש העתידי שלנו בה.

:Multi-Dimensional Scaling (MDS)

השיטות שמופיעות כאן נועדו לפתור את הבעיה הבאה:
נתון דאטא בצורת מטריצת מרחקים בין הנקודות, אבל לא ייצוג שלהן כאוסף וקטורים במרחב. המטרה היא לייצג אותם ככאלה, כאשר שומרים על המרחקים ככל האפשר או בהתאם למה שנדרוש. בכל מקרה, אלה שיטות שדורשות זמן חישוב די ארוך.

:Classical MDS (cMDS)

השיטה ידועה גם כ-Principal Coordinate Analysis (PCoA).
קלאסי – מחפשים ייצוג במרחב של הדאטא, כך שההפרשים בין המרחקים של נקודות במרחב למרחקים במטריצת המרחקים יהיו הקטנים ביותר האפשריים (כלומר, שנשמור על המרחקים שבין הנקודות גם כשנציג אותן בתור וקטורים).
דרך הפתרון (בקצרה): נרצה להטיל את הנקודות במרחב, למטריצה X (ראו הערה). נסתכל על המטריצה $B = X^T X$, ונציג אותה בשפה של המרחקים הידועים לנו. אחר כך נבצע SVD למטריצה: $B = UDV$ כאשר U ו- V מטריצות אוניטריות (במקרה הממשי, $UU^T = I$) ו- D מטריצה אלכסונית. בשלב הבא, $X = D^{\frac{1}{2}}U^T$.

הערה – ביצוע שלבים אלה בצורה פשוטה ייתנו ל- X מימד פיצ'רים ששווה לכמות הנקודות בדאטא, שזה הרבה. כדי למנוע זאת, משתמשים רק בערכים העצמיים הגדולים ביותר של D . אינטואיטיבית, זה אומר להיפטר מפיצ'רים שהערכים שלהם קרובים מאוד ל-0.

הבעיה באלגוריתם הזה היא שהוא לוקח זמן – SVD עולה $O(n^3)$ כאשר n כמות הנקודות.

:Metric MDS (mMDS)

הכללה מסוימת של cMDS, כך שהמרחקים בין הנקודות יהיו לא בהכרח לפי מטריקה אוקלידית (כלומר הייצוג החדש יתאים למטריצת מרחקים מועתקת למרחב חדש, או העתקה של הייצוג החדש מתאימה למטריצת המרחקים המקורית).
מה שעושים הוא כותבים פונקציית loss (מכונה Stress) שאותה ממזערים על מרחב הנקודות, באמצעות שיטות אופטימיזציה כמו גרדיאנט.

:Non-Metric MDS (nMDS)

כאן דורשים תנאי חלש יותר מקודם –
אם $d_{ij} < d_{kl}$ במטריצה המקורית, אז $d(x_i, x_j) \leq d(x_k, x_l)$ במרחב החדש לכמה שיותר נקודות.
דרך פתרון:

1. בחירת נקודות אקראיות מהתפלגות נורמלית בממד שנבחר מראש, עם תוחלת 0 ומטריצת Covariance אקראית.
2. לכל נקודה בדאטא תותאם נקודה מאוסף הנקודות הנ"ל.
3. לכל שתי נקודות, מחליפים ביניהן אם מספר הזוגות שעבורם התנאי מופר יקטן.

