APML
שיעור 11

Manifold Learning

איפה נשתמש בזה? ויזואליזציה, ארמון של הדאטא, preprocessing...

סימון: הדגמאות הן $x \in \mathbb{R}^p$, יש n דגמאות (קמח)

p הוא מימד גבוה מאוד. אז אינטואיטיבית נרצה מספר דגמאות

אקספוננציאלי ב-p כדי להבין את המרחב הזה. (n)

מימדים: ambient dimensionality – כמה פיצ'רים יש ישירות בדאטא?

למשל תמונה: מס' הפיקסלים × RGB

intrinsic dimensionality – כמה אינפורמציה באמת יש בדאטא?

כי תמונה בודדת ממש לא פורשת את

$\mathbb{R}^{30000}$ (גודל תמונה 100×100×3)

אפשר לחשוב על מבנה אחר של המימד של הדאטא.

⇐ נקבל שאת האינפורמציה שבדאטא אפשר

כנראה לייצג במספר משמעותית קטן יותר

של פיצ'רים.

אז כמו שהרבה פעמים מניחים תכונות שהדאטא כמו sparsity,

אותי נעסוק בדאטא שאפשר להניח שיש לו מימד אינטרינסי נמוך.

① דאטא ששוכן בתת מרחב לינארי ממימד נמוך.

נוכל להסביר את הדאטא למרחב הנמוך יותר, ועדיין לא

נאבד אינפורמציה (אינטואיטיבית תסביר אותנו בין הממדים)

⚠ אפשר לשלוח את x_i גם אז הדאטא כמעט משוכן בתת המרחב

ולא בדיוק משוכן בו.

איך נגלה מה המימד של הדאטא? (כמה המימד של המרחב בו הוא שוכן)

– נשים את הדאטא במטריצה $X_{p \times n}$, נדרג את המטריצה,

ונסתכל מה הדרגה של המטריצה המדורגת. – זו הדרגה שחיפשנו.

– לחלופין, נשתמש ב-SVD על אותה מטריצה $X_{p \times n}$, ומספר הערכים

הסינגולריים (איברי האלכסון של Σ ב- $U\Sigma V^T$) שאינם 0, זו הדרגה.

משהו שהיינו רוצים לשמר תחת הטלת הדאטא למימד אחר זה את

המרחקים בין צמדי הדאטא.

הוכחת מימד לבאוטאו שנוכן הוא נורמה לינארי - המשך

שיטה 1: PCA

מחשבים את מטריצת השונות האמפירית של הבאוטאו ($p \times p$):

$$S = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})(x_i - \bar{x})^T$$

Empirical Covariance Matrix

המטריצה S היא אומד קונסיסטנטי למטריצת הקווריאנס (population) Σ.

היא גם PSD, ולכן $S = U \Lambda U^T$ עבור U אורתוגונלית ו-Λ אלכסונית.

אז נעתיק את הבאוטאו שלנו למימד נמוך באופן הבא:

$$y_i = x_i \cdot U_d$$

data in new dimension d

data (as row vecs)

first d columns of U ($d \times p$)

שיטה 2: MDS

השיטה הזו אומרת רוצים לשמר את המרחקים בין הנתונים בבאוטאו

$$\Delta_{i,j} = \|x_i - x_j\|^2$$

ומבין נבנה הורדת מימד!

מי זה יקרה לנו? למשל כשמחפשים מרחק בין ספרים / ראיון בין אנשים על סמך חברים משותפים בפייסבוק / כדומה...

אז נגדיר מטריצת Similarity (חצי):

הפרגמה של המטריצה הזו היא d.

$$S = -\frac{1}{2} H \cdot \Delta \cdot H$$

כאשר $H = I - \frac{1}{n} \mathbb{1} \cdot \mathbb{1}^T$ מטריצה סימטרית שמאל data centering

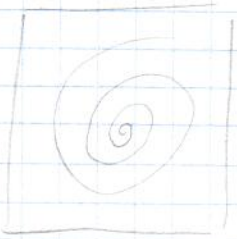
לאחר מכן מחשבים את הווקטורים מעל שורה בבאוטאו

נבחן את S: $S = U \Lambda U^T$ ונקבע את המטריצה $n \times d$

שעמודותיה הן $\sqrt{\lambda_i} u_i$ ($\lambda_1, \dots, \lambda_d$ איברי האלכסון של Λ)

ואז y_i הוא השורה ה-i של המטריצה הזו.

② דאטא שהוא Locally linear:



Swiss Roll Data

המבנה הגלובלי פה הוא לא לינארי.

במבנה כזה, המרחק האוקלידי ב-ambient space R^d

הוא לא אינפורמטיבי למרחק בין נקודות.

רוצים "לפרוש" את הדף הזה ולא להשתמש בישרות

במרחקים

APML שעור 12

Manifolds: מבחינה שני זה פשוט מבנה חלק (smooth) ורציף

כך שבנקודה מסוימת אפשר להעריך אותו לפי הסביבה הקרובה.

manifold ממימד d ניתן לקרב בנקודה מסוימת את מרחב לינארי ממימד d.

⇐ דאטא שיושב ב-Manifold

רוצים לקחת את $x_1, x_2, \dots, x_n$ הדאטא שלנו במימד גבוה לקואורדינטות

חדשות $y_1, \dots, y_n \in R^d$ כאשר הנחנו שהדאטא יושב ב-Manifold ממימד d.

נעסוק בשני אלגוריתמים: LLE / Diffusion maps

רוצים לשמר את המרחקים הקטנים $\|y_i - y_j\| \approx \|x_i - x_j\|$ if $\|x_i - x_j\| \le 1$

Locally Linear Embedding: LLE

הרעיון: נעשה הורדת מימד לינארית בכל נקודה, ואח"כ נשלב.

זה הגיוני במובן שאמרנו שה-Manifold מקרוב נראה לינארי.

(כמו שכדור הארץ נראה לנו שטוח ...)

האלגוריתם: 1. לכל נקודה x_i, נבנה neighbourhood של נקודות שקרובות

אליה במרחק אוקלידי ב-R^p. - מדיאלי kNN.

2. נבנה הורדת מימד לינארית לכל x_i בנפרד

3. נחבר אותם באופן חלק כלשהו.

הרעיון שלב 2: נחפש מטריצה W שמאזנת למינימום את ה residual sum of squares

עבור reconstruction של x_i מהשכנים שלו. → זה בעצם רגרסיה לינארית של x_i על השכנים שלו.

W הזו תהיה מאוד sparse ואפשר לחשוב עליה כמטריצת מעבר מ-x_i

על השכנים שלו (המתכנסים לנקודות קרובות 1-0 למי שלא ב-kNN)

$RRS(W) = \sum_i \|x_i - \sum_j W_{ij} x_j\|^2$

המשך: Manifold Learning

המשך: LLE

הגדרה שלם 3: איך נחפש את הנקודות אחרי הורדת המימד?

נמזער קואורדינטות Y שמתאימות למינימום את:

$$\Phi(Y) = \sum_{i=1}^{n} \left\| y_i - \sum_{i \neq j} W_{ij} y_j \right\|^2$$

כך ש- $Y^T Y = I$ ו- $\sum_j Y_{ij} = 0$

כדומר הפיצרים אורתוגונליים אחד לשני.

הערה חשובה:

* בחירת k שכנים הכי קרובים, אפשר יהיה לקבל E בשחו ולקחת את כל השכנים שבמרחק קטן מ E סביב הנקודה.

ב-kNN עלול להיכנס לנו נקודות רחוקות מאוד ובמרחק E אחרון לא יהיה כמה נקודות יהיו. זה הוא שקול בפאזה שהוא מפצה בצורה אחרת...

* איך עושים חיפוש kNN? מה הסיבוכיות?

הבעיה הגדולה זה $O(n^2)$ (n/d).

אפשר להשתמש ב- k-d tree (אם יש לנו את $x_1, \dots, x_n$ ולא רק מרחקים)

ויש גם אלגוריתמים לקירוב של kNN, שהם הרבה יותר יעילים.

רעיון נחמד: נעשה random-projection של הבאוסא ל-R^d

רוב הזמן זה טוב באופן מפתיע וזה בסיבוכיות הרבה יותר טובה (בעזרת FFT למשל)

נחפש k-NN במרחב המימד d

ואפשר לחזור על התהליך הזה כמה פעמים ולמצוא את המודלים בצורה חכמה בשבילו.

* אמרנו ש- W זה בעצם תוצאה של רגרסיה לינארית על כל נקודה, ולמעשה מקבלים קומבינציה אפינית לכל x_i.

* אפשר גם לדמיין מצב בו לא מקבלים את הבאוסא אלא רק מרחקים ומדיין נוכל למצוא הליך מקרי בין הנקודות

המשך: Manifold Learning

Diffusion Maps:

מתבסס על הרעיון של השרשרת המרקובית המאפשרת לדמות מרחקים בין האיברים בקבוצה אחרי זמן מסוים.

diffusion distance:

$$\Delta_{i,j}^{t} = \sqrt{\sum_{k=1}^{n} \frac{1}{d_k} \left(A_{ik}^{t} - A_{jk}^{t}\right)^2}$$

זה המרחק בין שני עצמים המתאפיינים בסימן i ובסימן j בזמן t-

העדכון מוביל את התיאור הנתונים באופן אמין.

נפרק את הווקטורים הנמוכים של A: $\Phi_1, \dots, \Phi_n$, המשמרים לפי $1 = \lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$.

ונגדיר:

$$\Phi_t : x_i \longrightarrow (\lambda_2^t \Phi_2(i), \dots, \lambda_n^t \Phi_n(i))$$

diffusion map

אפשרי נקבל מרחקים הנכונים!:

$$\|\Phi_t(x_i) - \Phi_t(x_j)\| = \Delta_{i,j}^{t}$$

מה זה המימדי לפי הקואורדינטות - מימד n.

ואפשר להוריד את המימד עוד כי זה שניקח מה רק את t הקואורדינטות הראשונות