L19: Linear Distance Metric Learning

Data Mining: Jeff M. Phillips

slides mostly by Meysam Alishahi

March 26, 2025

Data $X = \{x_1, x_2 \dots x_n\} \in \mathbb{R}^d$

like define distance $D(x_1, x_2)$

$$x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \\ x_{id} \end{bmatrix} \qquad \neq \|x_1 - x_1\|$$

Learn $\tilde{X} \leftarrow XA$

① MultiDimensional Scaling (MDS)

a distance D → low dim Euclidean

② Latent Discriminant Analysis (LDA)

k- clusters → low-dim Euclidean

③ Linear Distance Metric Learning (LDML)

pairs of close/far points → low-d Euclidean

Multidimensional Scaling

Input **Distance Matrix** $D \in \mathbb{R}^{n \times n}$

$$D_{ij} = D(x_i, x_j)$$

Output for dim k (eg. $k=2$)

Find $\psi(x_i) \rightarrow z_i \in \mathbb{R}^k$

$$D(x_i, x_j) \approx \| \psi(x_i) - \psi(x_j) \|^2$$

classic MDS

1. Convert $D \in \mathbb{R}^{n \times n}$ to $D^{(2)}$: $D^{(2)}_{ij} = (D_{ij})^2$

2. <u>Double Centering</u>

$$\begin{bmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{bmatrix} - \underbrace{\begin{bmatrix} 1/n & 1/n & 1/n \\ & \ddots & \\ & & 1/n \end{bmatrix}}_{n \times n}$$

centering matrix $C_n = I_n - \frac{1}{n}\mathbb{1}$

$$M = -\frac{1}{2} C_n D^{(2)} C_n$$

3. EigenDecomposition $[V, L] = \text{eigs}(M)$ A A^T

$$M = VLV^T = \underbrace{(VL^{1/2})}_{\text{data}}(VL^{1/2})^T$$

4. Project onto top k eigenvector,

return $B = \underbrace{V_k}_{\text{eigen vectors}} \underbrace{L_k^{1/2}}_{\text{scaling}} \in \mathbb{R}^{n \times k}$

Why does MDS work?

$AA^T = S \qquad S_{ij} = \langle a_i, a_j \rangle$

[illegible]

↳ eigs + top k ⇒ mapping $\mathbb{R}^k$

$$\boxed{\|a_i - a_j\|^2 \underline{=} D(x_i, x_j)^2} = \underline{\|a_i\|^2} + \underline{\|a_j\|^2} - 2\boxed{\langle a_i, a_j \rangle} = S_{ij}$$

assume in $\mathbb{R}^d$ $\quad a_1 = 0 \in \mathbb{R}^d$

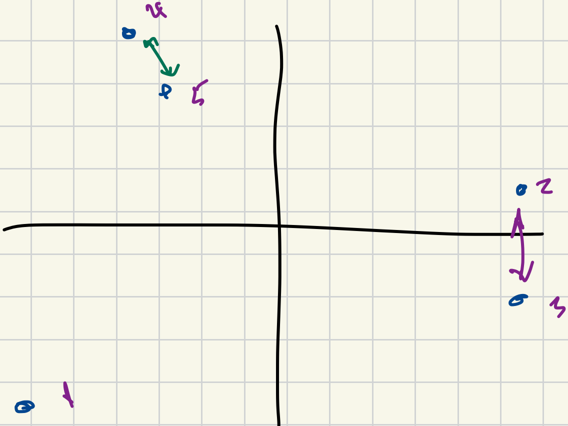
$$\|a_i\|^2 = \|a_i - 0\|^2 = D(x_i, x_1)^2$$

$$\langle a_i, a_j \rangle = -\frac{1}{2}\left(D(x_i, x_j)^2 - D(x_i, x_1)^2 - D(x_j, x_1)^2 \right)$$

arbitrary choice

$D_{2,3}$

$$D = \begin{bmatrix} 0 & 4 & 3 & 7 & 8 \\ 4 & 0 & \boxed{1} & 6 & 7 \\ 3 & 1 & 0 & 5 & 7 \\ 7 & 6 & 5 & 0 & 1 \\ 8 & 7 & 7 & 1 & 0 \end{bmatrix} \in \mathbb{R}^{5 \times 5} \qquad n = 5$$



Dimensionality Reduction for Visualization

Setting

- High-dimensional data $X \in \mathbb{R}^d$ with d large (e.g., $d = 1000$)
- Want best $k = 2$ representation, so can plot.

Common approaches:

- PCA - linear, minimizes squared error in projection
- t-SNE (and relatives) - non-linear, tries to preserve nearby-structure (perplexity)

Dimensionality Reduction for Visualization

Setting

- High-dimensional data $X \in \mathbb{R}^d$ with d large (e.g., $d = 1000$)
- Want best $k = 2$ representation, so can plot.

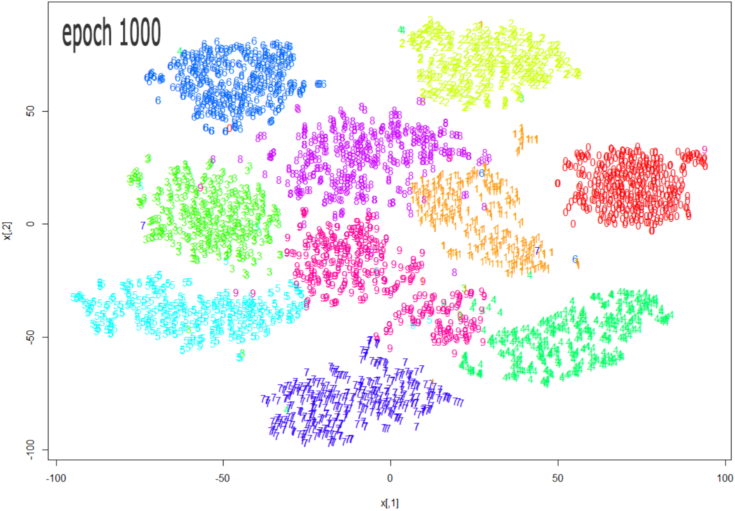
Common approaches:

- PCA - linear, minimizes squared error in projection
- t-SNE (and relatives) - non-linear, tries to preserve nearby-structure (perplexity)

Supervised Dimensionality Reduction:

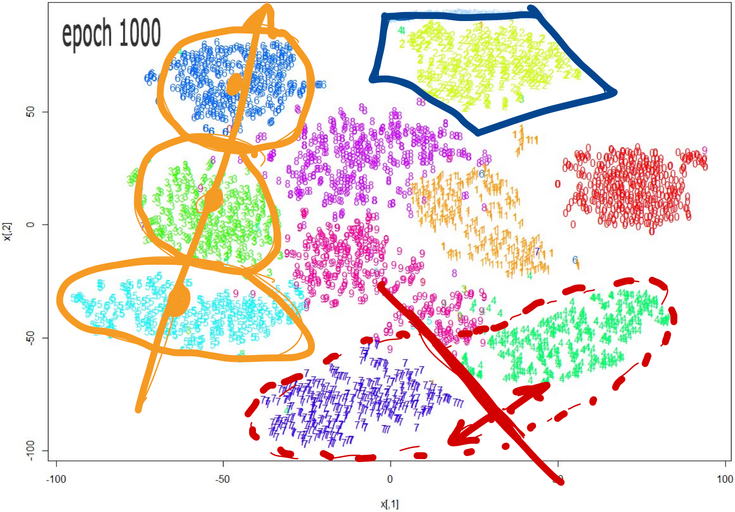
- Linear Discriminant Analysis (LDA) – "classic"
- **Linear** Distance Metric **Learning** – (JMLR 2024 w/ M. Alishahi, A Little)

My "beef" with t-SNE: (#1) Non-Linearity



Are linear separators, shapes (convex hulls), linear sequences real?

My "beef" with t-SNE: (#1) Non-Linearity



Are linear separators, shapes (convex hulls), linear sequences real?

My "beef" with t-SNE: (#1) Non-Linearity

Linear methods (like PCA) do ensure:

- linear separators seen in projection → exist in high-d
- shapes in projection → can be separated by convex hulls in high-d
- linear sequences → can be fit to linear patterns in high-d (may be deviations)

My "beef" with t-SNE: (#1) Non-Linearity

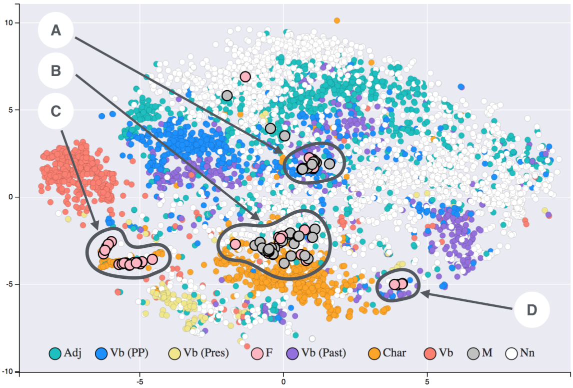
Linear methods (like PCA) do ensure:

- linear separators seen in projection → exist in high-d
- shapes in projection → can be separated by convex hulls in high-d
- linear sequences → can be fit to linear patterns in high-d (may be deviations)

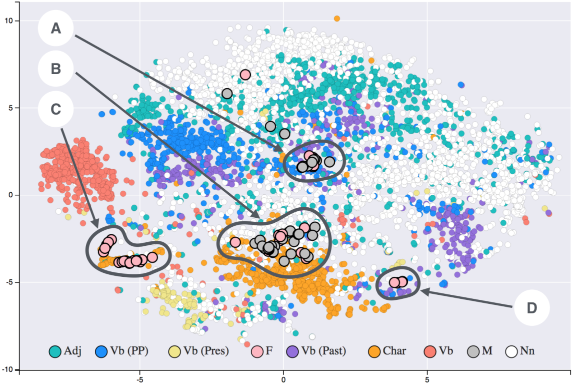
Also, linear methods are **generalizable** to new data.
Its a linear rule which we can apply to data not yet seen.

$$\hat{X} \leftarrow X \underbrace{A}_{\text{mapping}}$$

My "beef" with t-SNE: (#2) UN-Supervised

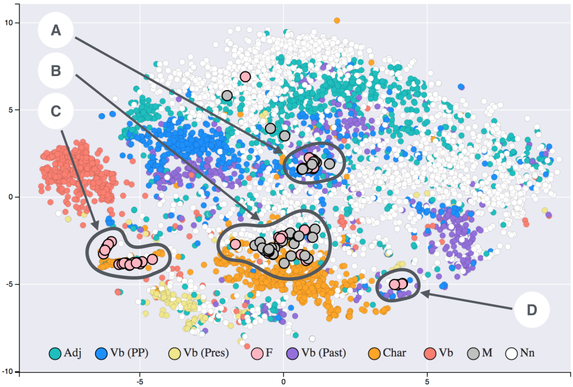


My "beef" with t-SNE: (#2) UN-Supervised



Sci: Cool! I did science.

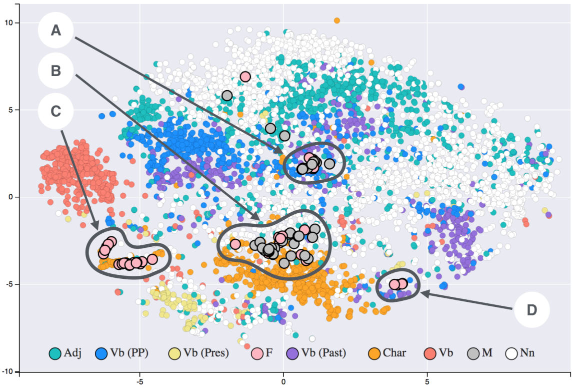
My "beef" with t-SNE: (#2) UN-Supervised



Sci: Cool! I did science.

JP: How do you know its good?

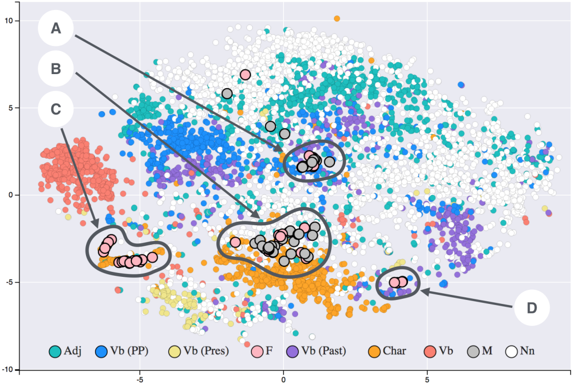
My "beef" with t-SNE: (#2) UN-Supervised



Sci: Cool! I did science.   JP: How do you know its good?

Sci: Oh, I measured on data I know?

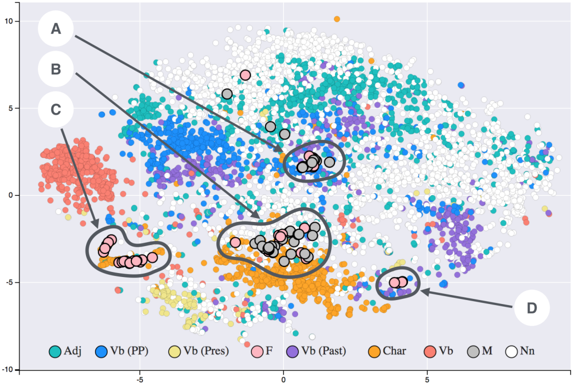
My "beef" with t-SNE: (#2) UN-Supervised



Sci: Cool! I did science. JP: How do you know its good?

Sci: Oh, I measured on data I know? JP: Wait, so you have labels, did you use them to train?

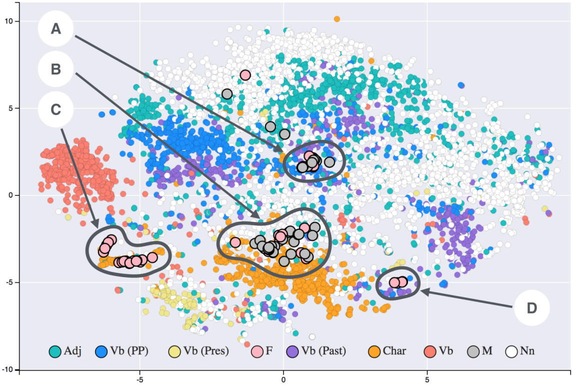
My "beef" with t-SNE: (#2) UN-Supervised



Sci: Cool! I did science. JP: How do you know its good?
Sci: Oh, I measured on data I know? JP: Wait, so you have labels,
did you use them to train?

Sci: No. ???

My "beef" with t-SNE: (#2) UN-Supervised

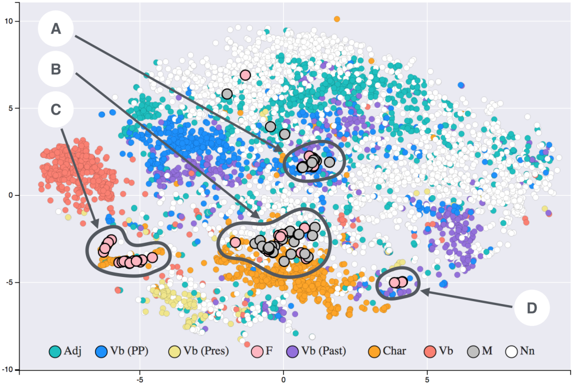


Sci: Cool! I did science. JP: How do you know its good?

Sci: Oh, I measured on data I know? JP: Wait, so you have labels, did you use them to train?

Sci: No. ??? JP: Why not?

My "beef" with t-SNE: (#2) UN-Supervised



Sci: Cool! I did science. JP: How do you know its good?

Sci: Oh, I measured on data I know? JP: Wait, so you have labels,
did you use them to train?

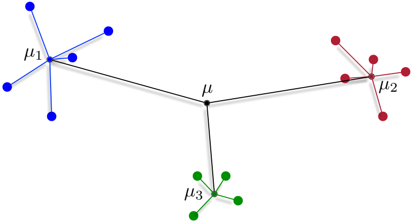
Sci: No. ??? JP: Why not?

Sci: Huh? What do you mean?

Linear (Fisher) Discriminant Analysis

Data $X \in \mathbb{R}^d$; each $x_i \in X$ has $y_i \in [k]$ (one of k classes)

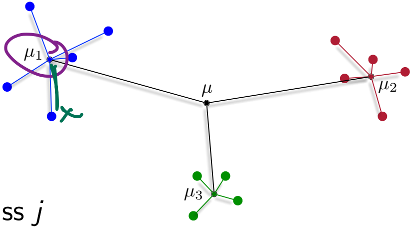
$S_j = \{x_i \in X \mid y_i = j\}$



Linear (Fisher) Discriminant Analysis

Data $X \in \mathbb{R}^d$; each $x_i \in X$ has $y_i \in [k]$ (one of k classes)

$S_j = \{x_i \in X \mid y_i = j\}$



$\underline{\mu_j} = \sum_j |S_j| \sum_{x \in S_j} x$ mean of class j

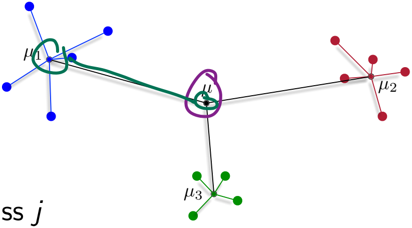
$\underline{\Sigma_j} = \frac{1}{|S_j|} \sum_{x \in S_j} \underline{(x - \mu_j)}\underline{(x - \mu_j)}^T$ covariance of j

within class covariance $\underline{\Sigma_W} = \underline{\frac{1}{|X|} \sum_{j=1}^{k} |S_j| \Sigma_j}$

Linear (Fisher) Discriminant Analysis

Data $X \in \mathbb{R}^d$; each $x_i \in X$ has $y_i \in [k]$ (one of k classes)

$S_j = \{x_i \in X \mid y_i = j\}$



$\mu_j = \sum_j |S_j| \sum_{x \in S_j} x$ mean of class j

$\Sigma_j = \frac{1}{|S_j|} \sum_{x \in S_j} (x - \mu_j)(x - \mu_j)^T$ covariance of j

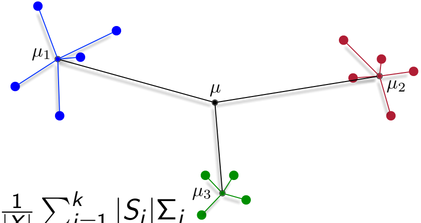
within class covariance $\Sigma_W = \frac{1}{|X|} \sum_{j=1}^{k} |S_j| \Sigma_j$

$\boxed{\mu = \frac{1}{|X|} \sum_{x \in X} x}$ overall mean

between class covariance $\Sigma_B = \frac{1}{|X|} \sum_{j=1}^{k} |S_j| \overline{(\underline{\mu_j} - \underline{\mu})}\overline{(\mu_j - \mu)}^T$

Linear (Fisher) Discriminant Analysis

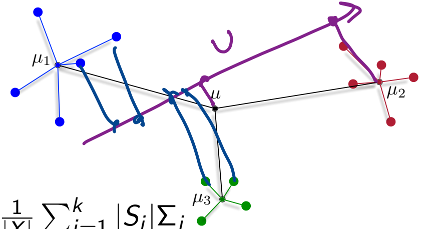
Data $X \in \mathbb{R}^d$; has $y_i \in [k]$
$S_j = \{x_i \in X \mid y_i = j\}$



within class covariance $\underline{\Sigma_W} = \frac{1}{|X|} \sum_{j=1}^{k} |S_j| \Sigma_j$
between class covariance $\underline{\Sigma_B} = \frac{1}{|X|} \sum_{j=1}^{k} |S_j| (\mu_j - \mu)(\mu_j - \mu)^T$

Linear (Fisher) Discriminant Analysis

Data $X \in \mathbb{R}^d$; has $y_i \in [k]$
$S_j = \{x_i \in X \mid y_i = j\}$



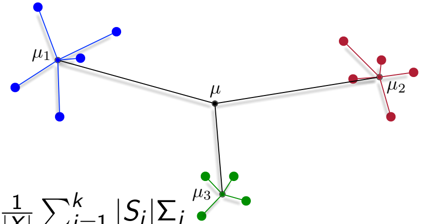
within class covariance $\Sigma_W = \frac{1}{|X|}\sum_{j=1}^{k} |S_j|\Sigma_j$
between class covariance $\Sigma_B = \frac{1}{|X|}\sum_{j=1}^{k} |S_j|(\mu_j - \mu)(\mu_j - \mu)^T$

Find direction u maximizing $\frac{u^T \Sigma_B u}{u^T \Sigma_W u}$

Linear (Fisher) Discriminant Analysis

Data $X \in \mathbb{R}^d$; has $y_i \in [k]$

$S_j = \{x_i \in X \mid y_i = j\}$



within class covariance $\Sigma_W = \frac{1}{|X|} \sum_{j=1}^{k} |S_j| \Sigma_j$

between class covariance $\Sigma_B = \frac{1}{|X|} \sum_{j=1}^{k} |S_j| (\mu_j - \mu)(\mu_j - \mu)^T$

Find direction u maximizing $\frac{u^T \Sigma_B u}{u^T \Sigma_W u}$

Let V_2 be top 2-eigenvectors of $\Sigma_W^{-1} \Sigma_B$

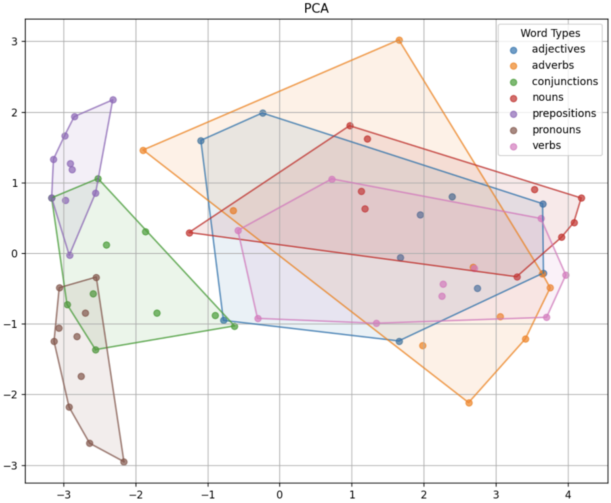
$\tilde{X} \Leftarrow V_2^T X$ (points in $2d$)

Embeddings by Word Type

Embed 70 words via GloVE in $d = 100$: 10 each of ... nouns, verbs, adjectives, adverbs, conjunctions, prepositions, pronouns

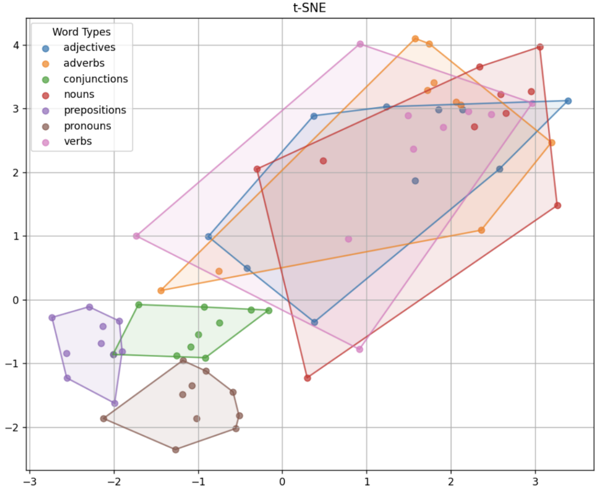
Embeddings by Word Type

Embed 70 words via GloVE in $d = 100$: 10 each of ... nouns, verbs, adjectives, adverbs, conjunctions, prepositions, pronouns



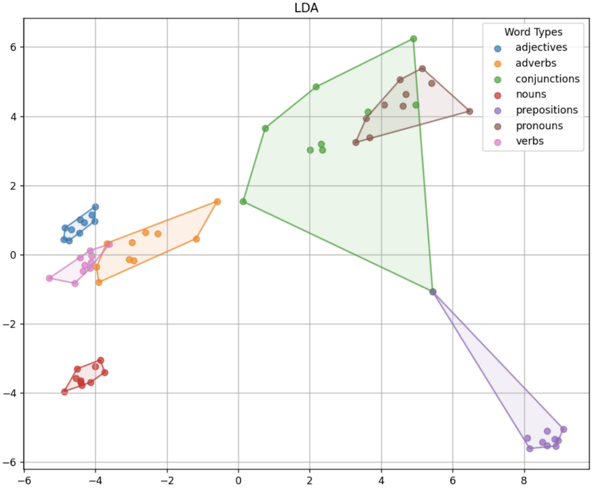
Embeddings by Word Type

Embed 70 words via GloVE in $d = 100$: 10 each of ... nouns, verbs, adjectives, adverbs, conjunctions, prepositions, pronouns



Embeddings by Word Type

Embed 70 words via GloVE in $d = 100$: 10 each of ... nouns, verbs, adjectives, adverbs, conjunctions, prepositions, pronouns



Goal of Linear DML

Setting

- Data $X \in \mathbb{R}^d$ with an underlying metric $d_E(x, p) = \|x - p\|$
- We do not trust $d_E(x, y) = \|x - y\|$.
- Given pairs $\{(x_1, x_1'), (x_2, x_2'), ..., (x_m, x_m')\}$ each with label $y_i \in \{\text{Similar, Dissimilar}\}$

Goal:

- map the data into a metric space so that the distance between points in the second space optimizes similarity and dissimilarity information provided within the data.

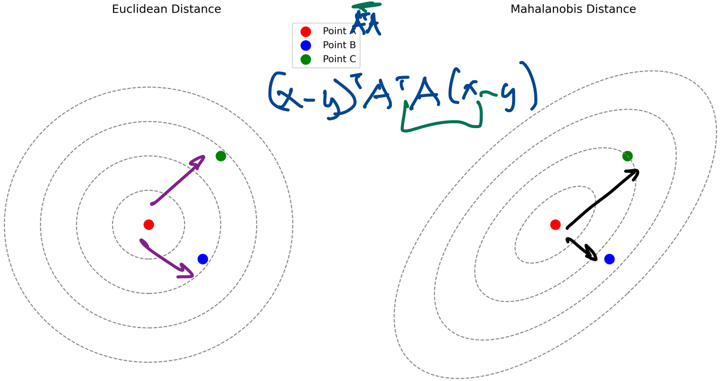
Many studies focused on **non-linear (NN based)** mappings. We only consider **linear** mappings.

$\tilde{X} = AX$

$M = A^T A$

Mahalanobis distance $d_M(x, y) = \sqrt{(x-y)^T M (x-y)}$, where M is p.s.d matrix.

$\overline{A^T A}$



$(x-y)^T A^T \underbrace{A(x-y)}$

Unit ball $\{x \in \mathbb{R}^d \mid d_M(x, p) \leq 1\}$ is ellipsoid.

Why Mahalanobis Distance for Linear Distance Metric Learning?

- **Captures Affine Transformations:** Any linear transformation $x \to Ax$ can be captured by the Mahalanobis distance by $M = A^\top A$:

$$\|Ax - Ay\|^2 = \|x - y\|^2_{A^\top A}$$

Why Mahalanobis Distance for Linear Distance Metric Learning?

- **Captures Affine Transformations:** Any linear transformation $x \to Ax$ can be captured by the Mahalanobis distance by $M = A^\top A$:

$$\|Ax - Ay\|^2 = \|x - y\|^2_{A^\top A}$$

  - **Scaling:** Accounts for varying feature scales.
  - **Rotation:** Captures dependencies between features (non-axis-aligned metrics).
  - **Linear Structure:** It preserves linear structure.

- Looking for M reflecting similarities and dissimilarities.

- Looking for M reflecting similarities and dissimilarities.

- **But how can we appropriately formulate this problem?**

DML-eig method (Ying and Li (2012))

(In mathfordata.github.io)

Maximizes the minimum distance between dissimilar pairs while constraining sum of similarities within a bound.

$$\max_{M \succeq 0} \min_{(x_i,x_j)\in\mathcal{D}} d_M^2(x_i,x_j) \quad \text{min close pairs}$$

$$\text{s.t.} \sum_{(x_i,x_j)\in\mathcal{S}} d_M^2(x_i,x_j) \le 1. \quad \text{restriction}$$

Σ for pairs $\le \underline{1}$

DML-eig method (Ying and Li (2012))

(In mathfordata.github.io)

Maximizes the minimum distance between dissimilar pairs while constraining sum of similarities within a bound.

$$\max_{M \succeq 0} \min_{(x_i, x_j) \in \mathcal{D}} d_M^2(x_i, x_j)$$
$$\text{s.t.} \sum_{(x_i, x_j) \in \mathcal{S}} d_M^2(x_i, x_j) \leq 1.$$

Properties:

- Reduced to eigenvalue optimization framework
- Subgradient ascent optimization approach avoids projection but still requires an $O(d^3)$ eigendecomposition step.
- Outperforming other baselines in experimental evaluations.

Model Assumptions

- **Data Setup:**

Model Assumptions

- **Data Setup:**
  - We are given N iid observations $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}^d$ and each pair is given a label $\ell_i \in \{\mathrm{Far}, \mathrm{Close}\}$.

Model Assumptions

- **Data Setup:**
  - We are given N iid observations $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}^d$ and each pair is given a label $\ell_i \in \{\mathrm{Far}, \mathrm{Close}\}$.
- **Label Generation Assumptions:**

Model Assumptions

- **Data Setup:**
  - We are given N iid observations $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}^d$ and each pair is given a label $\ell_i \in \{\mathrm{Far}, \mathrm{Close}\}$.
- **Label Generation Assumptions:**
  - There are p.s.d. $\underline{M^*} \in \mathbb{R}^{d \times d}$ and a threshold $\underline{\tau^*}$ which generates labels $\ell_i \in \{\mathrm{Close}, \mathrm{Far}\}$.
  - The pair (x_i, y_i) is labeled **Close** if and only if

$$\|x_i - y_i\|^2_{M^*} + \underline{\eta_i} < \underline{\tau^*}, \tag{Label Assumption}$$

    where $\eta_i \sim \mathrm{Noise}(\eta | 0, s)$ is a noise term.
  - Noise η_i is iid and follows a distribution $\mathrm{Noise}(\eta | 0, s)$.

Model Assumptions

- **Data Setup:**
  - We are given N iid observations $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}^d$ and each pair is given a label $\ell_i \in \{\mathrm{Far}, \mathrm{Close}\}$.
- **Label Generation Assumptions:**
  - There are p.s.d. $M^* \in \mathbb{R}^{d \times d}$ and a threshold τ^* which generates labels $\ell_i \in \{\mathrm{Close}, \mathrm{Far}\}$.
  - The pair (x_i, y_i) is labeled **Close** if and only if

$$\|x_i - y_i\|_{M^*}^2 + \eta_i < \tau^*, \qquad \text{(Label Assumption)}$$

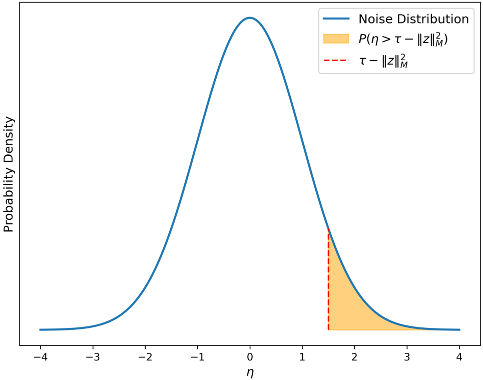
    where $\eta_i \sim \mathrm{Noise}(\eta|0, s)$ is a noise term.
  - Noise η_i is iid and follows a distribution $\mathrm{Noise}(\eta|0, s)$.

Note: The labeling is probabilistic due to the noise η

Optimal Loss Functions

- **Setup:** $\ell =$ **Far** if and only if $\eta > \tau - \|x - y\|_M^2$,
- **Labeling Distribution:** for $z = x - y$,

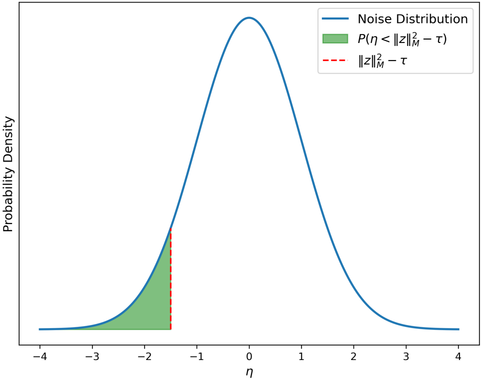
$$P(\ell = 1 | z; M, \tau) = \mathrm{Pr}(\underline{\eta} > \underline{\tau - \|z\|_M^2}),$$



Optimal Loss Functions

- **Setup:** $\ell =$ **Far** if and only if $\eta > \tau - \|x - y\|_M^2$,
- **Labeling Distribution:** for $z = x - y$,

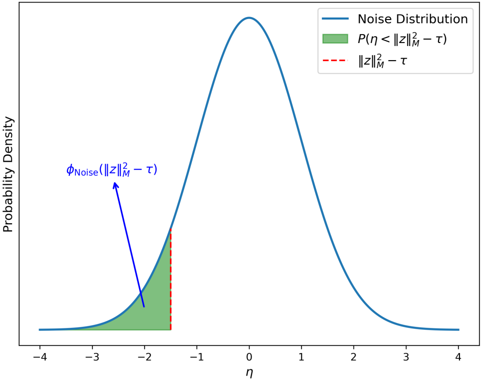
$$P(\ell = 1 | z; M, \tau) = \Pr(\eta > \tau - \|z\|_M^2),$$



Optimal Loss Functions

- **Setup:** $\ell =$ **Far** if and only if $\eta > \tau - \|x - y\|_M^2$,
- **Labeling Distribution:** for $z = x - y$,

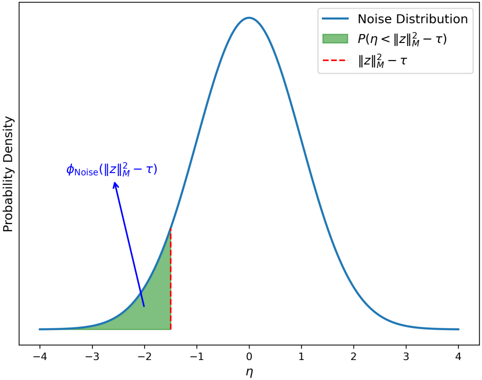
$$P(\ell = 1|z; M, \tau) = \mathrm{Pr}(\eta > \tau - \|z\|_M^2),$$



Optimal Loss Functions

- **Setup:** $\ell =$ **Far** if and only if $\eta > \tau - \|x - y\|_M^2$,
- **Labeling Distribution:** for $z = x - y$,

$$P(\ell = 1 | z; M, \tau) = \Phi_{\text{Noise}}(\|z\|_M^2 - \tau)$$



Optimal Loss Functions

- **Maximum Likelihood Estimation (MLE):**
  - Objective: Find a p.s.d. matrix M and threshold τ that minimize the **empirical risk**:

$$\boxed{R_N(M,\tau) = -\frac{1}{N}\sum_{i=1}^{N} \log \Phi_{\mathrm{Noise}}\left(\ell_i(\|z_i\|_M^2 - \tau)\right).}$$

← GD

- **True Risk Function:**

$$R(M,\tau) = -\mathop{\mathbb{E}}_{z,\ell} \log \Phi_{\mathrm{Noise}}\left(\ell(\|z\|_M^2 - \tau)\right).$$

- **Optimization problem:**

$$\min_{M\succeq 0,\tau\geq 0} R_N(M,\tau),$$

Approximation Guarantees

- **Convex Optimisation:**
  - Both $R(M, \tau)$ and $R_N(M, \tau)$ are convex.
  - $R(M, \tau)$ is uniquely minimised at (M^*, τ^*).
- **Convergence and Error Bounds:**
  - $R_N(M, \tau)$ converges uniformly to $R(M, \tau)$.
  - If $N > N_d(\varepsilon, \delta) = O(\frac{1}{\varepsilon^2}(\log \frac{1}{\delta} + d^2 \log \frac{d}{\varepsilon}))$, then with probability at least $1 - \delta$,

$$\sup_{(M,\tau)} |R_N(M, \tau) - R(M, \tau)| < \varepsilon.$$

  - The minimizer $(\hat{M}, \hat{\tau}) \in \operatorname{argmin} R_N(M, \tau)$ satisfies:

$$0 < R(\hat{M}, \hat{\tau}) - R(M^*, \tau^*) < \varepsilon$$

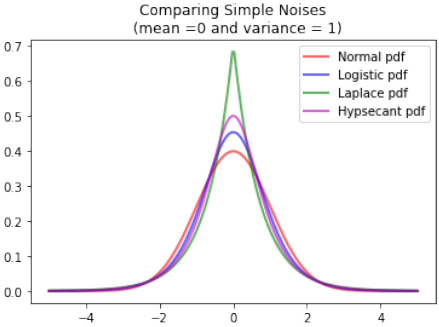
Recovery Guarantees

- **Parameter Recovery:**
  - Under some constraints:

$$\|M^* - \hat{M}\|_2 + |\tau^* - \hat{\tau}| \leq \varepsilon$$

  - Holds for sufficiently large $N > N(\frac{\varepsilon^2}{c^d}, \delta)$.
- **Low-Rank Model: (First Provable Guarantee)**
  - Our method supports truncating the learned $\hat{M}$ to a low-rank matrix while preserving accuracy in loss and parameters.
- **Empirical Success:**
  - Achieves high accuracy (over 99%) and precise parameter recovery (multiplicative error < 1.01) on noiseless, noisy, synthetic, and real data.
  - Robust to mislabeled data (e.g., accurately recovers true parameters with 45% mislabeled data).

Different noise options



Logistic Distribution as the Noise

Closed form for

$$\Phi_{\mathrm{Noise}}(t) = \sigma(t) = \frac{1}{1 + e^{-t}}$$

Experimental Results

Data Generation Process

- Randomly generate $\Sigma_{d\times d}$ as the covariance matrix.
- Independently sample $2N$ points (x_i, y_i) from $\mathcal{N}(\mathbf{0}, \Sigma)$.
- Generate N pairs (x_i, y_i), $i = 1, \ldots, N$.
- Randomly construct M^* as a p.s.d. matrix of rank $k \leq d$.
- Randomly select $\tau^* > 0$ (close the value making balanced labeling)

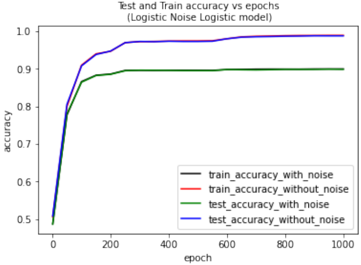
Noisy Labeling Scenarios

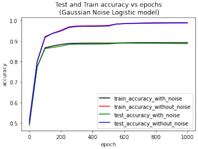
- Random noise $\eta_i \sim$ Noise(0,s) (e.g., Gaussian) with scale s
-

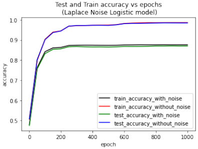
$$\ell_i = \textbf{Far} \qquad \text{if and only if} \qquad \|x_i - y_i\|^2_{M^*} + \eta_i \geq \tau^*$$

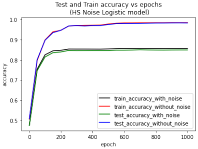
Logistic model with different noises: accuracy VS epochs

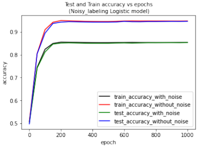
$d = 10$, and the noise results in a misclassification rate of 10%.











Noise type:	Logistic	Gaussian	Laplace	HS	Noisy Labeling
train acc. w/ noise	89.93% (0.22)	89.51% (0.20)	87.35% (0.28)	85.48% (0.30)	85.73% (0.34)
train acc. w/o noise	98.80% (0.10)	98.79% (0.19)	98.61% (0.13)	98.53% (0.14)	94.68% (0.32)
test acc. w/ noise	89.76% (0.40)	89.34% (0.32)	87.27% (0.51)	85.28% (0.47)	85.57% (0.65)
test acc. w/o noise	98.83% (0.21)	98.82% (0.18)	98.52% (0.21)	98.47% (0.23)	94.51% (0.60)

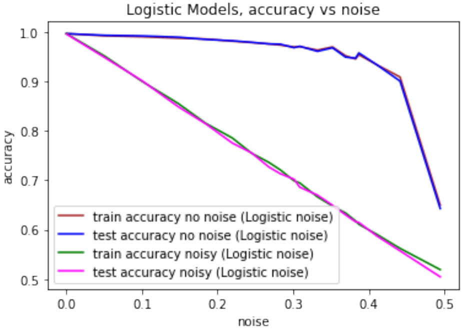
Table: Logistic model average accuracy (std) with different noise types (average over 20 trails) with 10% misclassifications labels.

Model Performance Summary with 10% noisy labeling

- The Logistic model accurately learns the labeling function:
  - **Noisy Labels:** $\sim 90\%$ accuracy (maximum possible with 10% noisy misclassification).
  - **Ground Truth Labels:** $\sim 99\%$ accuracy.
- Consistent performance on training and test data indicates no overfitting.
- Accuracy declines as noise deviates from the Logistic model (Gaussian, Laplace, Hyperbolic Secant, Noisy Labeling).
- The largest accuracy drop ($\sim 5\%$) occurs in the "noisy labeling" (change the true labeling directly) setting.

How Much Noise Can Break the Model?

- **Theoretical Insight:** Ground truth labeling recovery even with noisy labels.
- **Experimental Evidence:** Robustness with 10% mislabeling.
- **Procedure:** Training set size $= 15,000$ and $d = 10$. Gradually increase the misclassification rate and log accuracy.



How Much Noise Can Break the Model?

- **Observations:**
  - **Noisy Labels:**
    - Accuracy aligns with $y = 1 - x$ line (as expected).
  - **Ground Truth Labels:**
    - Robustness persists even with high noise.
    - 40% mislabeling yields 95% accuracy on unseen data.
    - Model starts to collapse when 45% labels are disturbed.
    - At 50% mislabeling, model still achieves 65% accuracy (train/test).
- **Conclusion:**
  - Despite high noise, extreme examples provide enough signal for the model to perform better than random guessing.

Sample Complexity in High Noise Setting

- **Impact of Noise Scale:**
  - Accuracy drops as noise increases, but theory predicts recovery with sufficient samples.
  - At 50% mislabeling with $15,000$ training samples, test accuracy drops to 65%
  - **Theory:** Each model can recover ground truth labeling regardless of noise level, given enough samples.

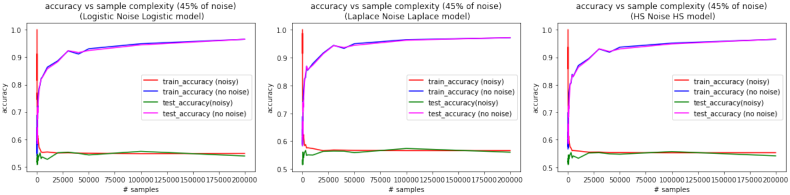


Figure: Accuracy VS Sample complexity with 45% noise when loss and noise are compatible.

Sample Complexity in High Noise Setting

- **Focus:** Examining sample complexity when loss and noise are compatible:
  - Logistic model for Logistic noise.
  - Laplace model for Laplace noise.
  - Hyperbolic Secant (HS) model for HS noise.
- **Experimental Evidence:**
  - At 50% mislabeling with $15,000$ training samples, test accuracy drops to 65%.
  - Increasing training samples to 2×10^5 improves accuracy
  - With 45% mislabeling, accuracy approaches 97%
- **Conclusion:**
  - Results validate theoretical predictions: Larger datasets mitigate noise effects and allow recovery of ground truth labeling.

Comparing to DML-eig

- **DML-eig Framework:**
  - DML-eig (Ying and Li (2012)) learns a Mahalanobis metric by optimizing eigenvalues.
  - Objective: Maximize minimal squared distances for dissimilar pairs, keeping similar pairs' squared distances bounded.
- **Comparison Setup:**
  - Use synthetic data with 0% and 10% noise.
  - Evaluate test accuracy on noisy and ground truth labels.

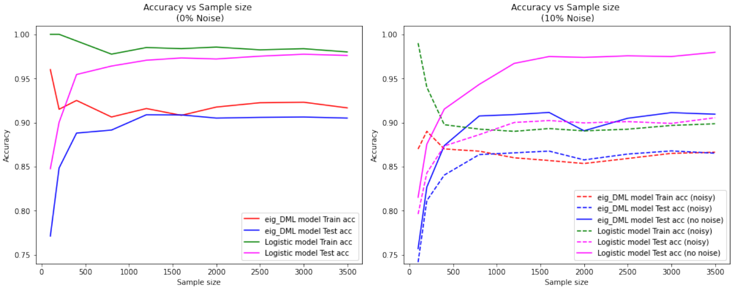


Figure: Performance of DML-eig with/without noise vs sample complexity

Comparing to DML-eig

- **Accuracy Results:**
  - **LDML:**
    - Achieves near 100% accuracy with sufficient data.
    - Robust to noise: Matches noisy training data and recovers ground truth labeling.
  - **DML-eig:**
    - Peaks at 90% accuracy in the noiseless setting.
    - Under noisy settings, achieves 85% test accuracy for noisy labels.
- **Scalability Results:**
  - **LDML:** Processes up to $10,000$ samples in 17 seconds, reaching 99% test accuracy.
  - **DML-eig:** Requires over 3 hours for $10,000$ samples, achieving only $85\% - 90\%$ accuracy.
- **Conclusion:**
  - LDML outperforms DML-eig in accuracy and scalability.
  - DML-eig struggles with noise and becomes computationally intractable with large sample sizes.

Unbalanced Labeling

- **Objective:** Study model robustness on unbalanced datasets.
- **Setup:** Gradually increase τ^* (30 values) to increase the Far label ratio.
- Total pairs: $60,000$ (with $20,000$ for testing).
- **Results:**
  - Overall accuracy remains high, regardless of label imbalance.
  - Accuracy for Far pairs drops to 93% at worst.
  - Accuracy for Close pairs drops to 78% at worst.
  - When Close pairs are $10\% - 98\%$, over 90% accuracy on all

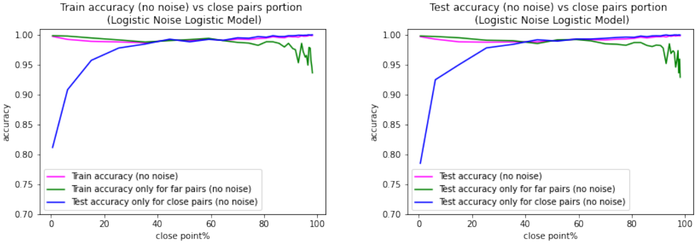


Figure: Performance of Logistic noise Logistic model on unbalanced data.

Unbalanced Labeling

- **Objective:** Study model robustness on unbalanced datasets.
- **Setup:** Gradually increase τ^* (30 values) to increase the Far label ratio.
- Total pairs: $60,000$ (with $20,000$ for testing).
- **Results:**
  - Overall accuracy remains high, regardless of label imbalance.
  - Accuracy for Far pairs drops to 93% at worst.
  - Accuracy for Close pairs drops to 78% at worst.
  - When Close pairs are $10\% - 98\%$, over 90% accuracy on all

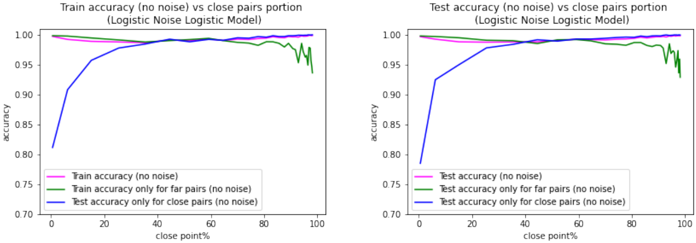


Figure: Performance of Logistic noise Logistic model on unbalanced data.

Unbalanced Labeling

- **Objective:** Study model robustness on unbalanced datasets.
- **Setup:** Gradually increase τ^* (30 values) to increase the Far label ratio.
- Total pairs: $60,000$ (with $20,000$ for testing).
- **Results:**
  - Overall accuracy remains high, regardless of label imbalance.
  - Accuracy for Far pairs drops to 93% at worst.
  - Accuracy for Close pairs drops to 78% at worst.
  - When Close pairs are 10% – 98%, over 90% accuracy on all

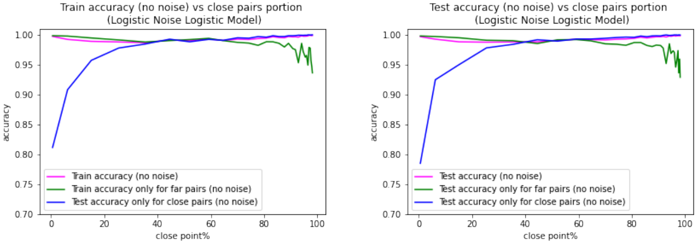


Figure: Performance of Logistic noise Logistic model on unbalanced data.

Instead of DML

Can we just normalize coordinate?

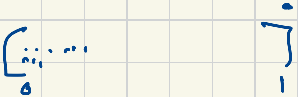
height
weight $\begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}$ x_i

height
$\{x_{11}, x_{21}, x_{31}, \ldots, x_{n1}\}$

- Set so $\underset{i}{\text{mean}}\, x_{ij} = 0$, $\sigma = 1$

- Set so $\underset{i}{\min}\, x_{ij} = 0$ $\underset{i}{\max}\, x_{ij} = 1$

Pitfalls



- if outliers, new data
  → change mapping.

- standardization assumes unimodal.

- if some coords are colinear

LDML Pros

① - works better if you first normaliz

② Generally better than nothing