Machine learning lecture slides

Multivariate Gaussians and PCA

Outline

- Multivariate Gaussians
- Eigendecompositions and covariance matrices
- Principal component analysis
- Principal component regression and spectral regularization
-
-

- Start with $X = (X_1, \ldots, X_d) \sim \mathrm{N}(0, I)$, i.e., $X_1, \ldots, X_d$ are iid $\mathrm{N}(0,1)$ random variables.
  - Probability density function is product of (univariate) Gaussian densities
  - $(X_i) = 0$

$/ \; j$

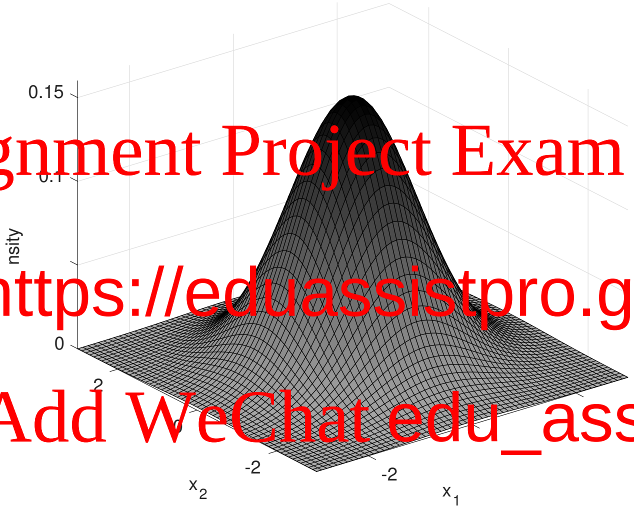


Figure 1: Density function for isotropic Gaussian in $\mathbb{R}^2$

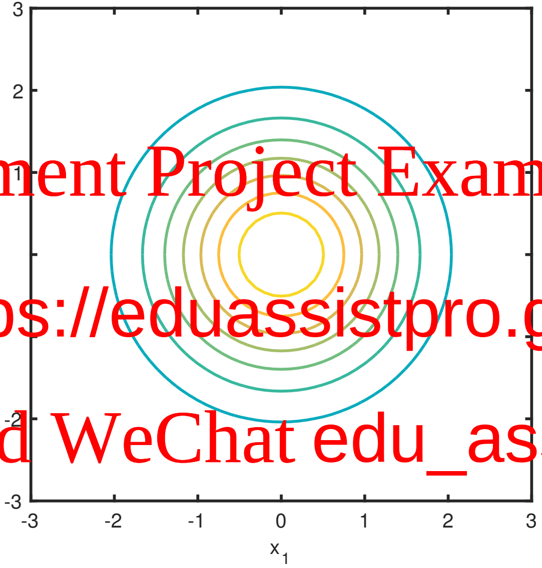


Figure 2: Density function level sets for isotropic Gaussian in $\mathbb{R}^2$

- Start with any random vector Z, then apply linear transformation, followed by translation
  - $X := MZ + \mu$, for $M \in \mathbb{R}^{k \times d}$ and $\mu \in \mathbb{R}^k$
  - Fact: $\mathbb{E}(X) = M\mathbb{E}(Z) + \mu$, $\operatorname{cov}(X) = M \operatorname{cov}(Z) M^{\intercal}$
  - E.g., let u ^{d} be a unit vector (u $_{2} = 1$), and $X := u^{\intercal} Z$

    (Z), and

- distribution, not just Gaussian distributi
  - However, it is convenient to illustrate the transformations on Gaussian distribu the Gaussian pdf is easy to understand.

Multivariate Gaussians: General Gaussians

- If $Z \sim \mathrm{N}(0, I)$ and $X = MZ + \mu$, we have $\mathbb{E}(X) = \mu$ and $\mathrm{cov}(X) = MM^\intercal$
  - Assume $M \in \mathbb{R}^{d \times d}$ is invertible (else we get a degenerate Gaussian distribution).
  - We say $X \quad \mathrm{N}(\mu, MM^\intercal)$

$$\ldots (x-\mu)\Big).$$

- Note: every non-singular covariance mat $MM^\intercal$ for some non-singular matrix

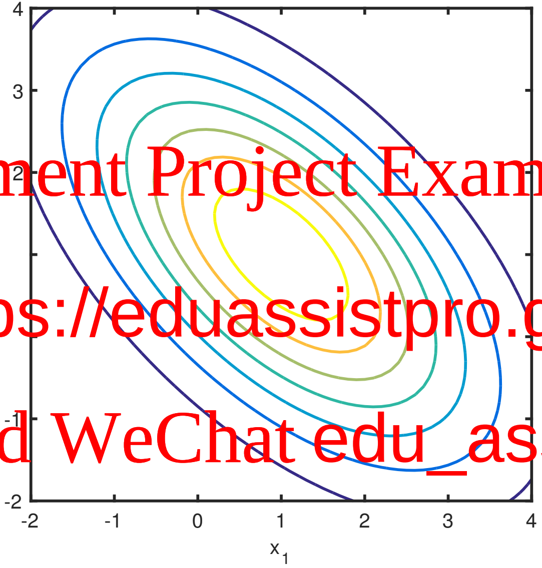


Figure 3: Density function level sets for anisotropic Gaussian in $\mathbb{R}^2$

Inference with multivariate Gaussians (2)

- Bivariate case: $(X_1, X_2) \sim \mathrm{N}(\mu, \Sigma)$ in $\mathbb{R}^2$

$$\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \quad \Sigma = \begin{bmatrix} \Sigma_{1,1} & \Sigma_{1,2} \\ \Sigma_{2,1} & \Sigma_{2,2} \end{bmatrix}$$

-
-
    - Miracle 1: it is a Gaussian distribution
    - Miracle 2: mean provided by linear predic with smallest MSE
    - Miracle 3: variance doesn't depend on

- What is the distribution of $X_2 \mid X_1 = x_1$?
  - Miracle 1: it is a Gaussian distribution
  - Miracle 2: mean provided by linear prediction of X_2 from X_1 with smallest MSE
  - Miracle 3: variance doesn't depend on x_1

e:

$$\hat{m} = \frac{\quad X_1, X_2}{\mathrm{var}(X_1)} \qquad \frac{\Sigma_{1,2}}{\quad}$$

$$\hat{\theta} = \mathbb{E}(X_2) - \hat{m}\mathbb{E}$$

  - Therefore:

$$\begin{aligned} \mathbb{E}[X_2 \mid X_1 = x_1] &= \hat{m}x_1 + \hat{\theta} \\ &= \mu_2 + \hat{m}(x_1 - \mu_1) \\ &= \mu_2 + \frac{\Sigma_{1,2}}{\Sigma_{1,1}}(x_1 - \mu_1) \end{aligned}$$

- What is the distribution of $X_2 \mid X_1 = x_1$?
  - Miracle 1: it is a Gaussian distribution
  - Miracle 2: mean provided by linear prediction of X_2 from X_1 with smallest MSE
  - Miracle 3: variance doesn't depend on x_1

$$])$$

$$= \Sigma_{2,2}$$

$$= \Sigma_{2,2}$$

$$= \Sigma_{2,2} - \frac{}{\Sigma_{1,1}^2} \Sigma_{1,1}$$

$$= \Sigma_{2,2} - \frac{\Sigma_{1,2}^2}{\Sigma_{1,1}}.$$

Inference with multivariate Gaussians (4)

- Beyond bivariate Gaussians: same as above, but just writing things properly using matrix notations

$$\mathbb{E}[X_2 \mid X_1 = x_1] = \mu_2 + \Sigma_{2,1}\Sigma_{1,1}^{-1}(x_1 - \mu_1)$$

1

- Every symmetric matrix $M \in \mathbb{R}^{d \times d}$ has d real *eigenvalues*, which we arrange as

$$\lambda_1 \geq \cdots \geq \lambda_d$$

-

  - This means

$$M v_i$$

for each $i = 1, \ldots, d$, and

$$v_i^{\top} v_j = \mathbf{1}_{\{i=j\}}$$

Eigendecomposition (2)

- Arrange $v_1, \ldots, v_d$ in an *orthogonal matrix* $V := [v_1 | \cdots | v_d]$
  - $V^\top V = I$ and $VV^\top = \sum_{i=1}^{d} v_i v_i^\top = I$
  - Therefore,

$$M = MVV^\top$$

$$d$$

$$\lambda_i v_i v_i$$

$$i=$$

- This is our preferred way to express the *m*
  - Also called *spectral decompositio*
  - Can also write $M = V\Lambda V^\top$, where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_d)$
  - The matrix V *diagonalizes* M:

$$V^\top M V = \Lambda$$

- $A \in \mathbb{R}^{n \times d}$ is data matrix
- $\Sigma := A^\mathsf{T} A = \frac{1}{n} \sum_{i=1}^{n} x_i x_i^\mathsf{T}$ is *(empirical) second-moment matrix*
  - If $\frac{1}{n} \sum_{i=1}^{n} x_i = 0$ (data are "centered"), this is the

-

$$u^\mathsf{T} \Sigma u = \frac{1}{n} \sum_{i=}^{n}$$

is (empirical) variance of data along direction u

Covariance matrix (2)

- Note: some pixels in OCR data have very little (or zero!) variation
  - These are "coordinate directions" (e.g., $u = (1, 0, \dots, 0)$)
  - Probably can/should ignore these!

Figure 4: Which pixels are likely to have very little variance?

- Σ is symmetric, so can write eigendecomposition

$$\Sigma = \sum_{i=1}^{d} \lambda_i v_i v_i^{\mathsf{T}}$$

-
-
  - This follows from the following charact

$$v_1^{\mathsf{T}} \Sigma v_1 = \max_{u \in \mathbb{R}^d : \|u}$$

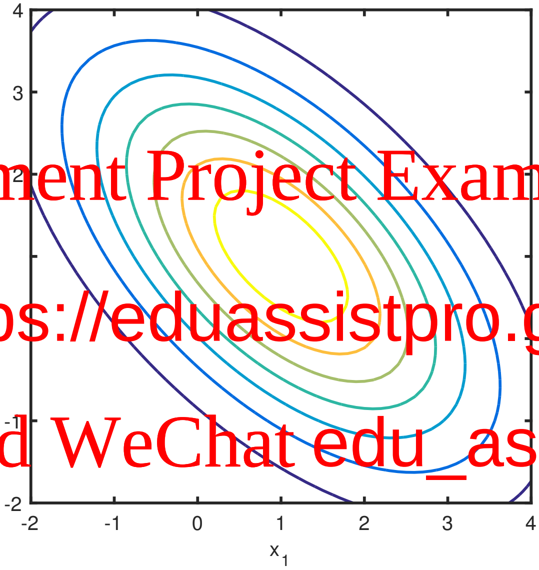


Figure 5: What is the direction of the top eigenvector for the covariance of this Gaussian?

- What about among directions orthogonal to v_1?
  - Answer: v_2, corresponding to second largest eigenvalue λ_2
- (Note: all eigenvalues of Σ are non-negative!)
- For any k, $V_k := [v_1 | \cdots | v_k]$ satisfies

$$\sum_{i=1}^{k} \lambda_i$$

(the *top k eigenvectors*)

- k-dimensional *principal components analysis (PCA)* mapping:

$$\varphi(x) = (x^\top v_1, \ldots, x^\top v_k) = V_k^\top x \in \mathbb{R}^k$$

where $V_k = [v_1 \cdots v_k] \in \mathbb{R}^{d\times k}$

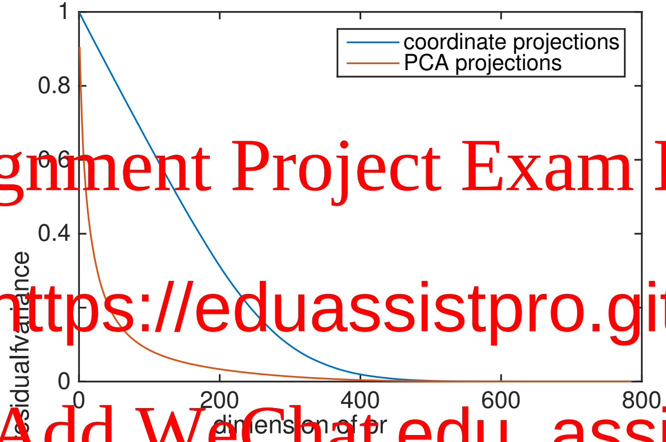


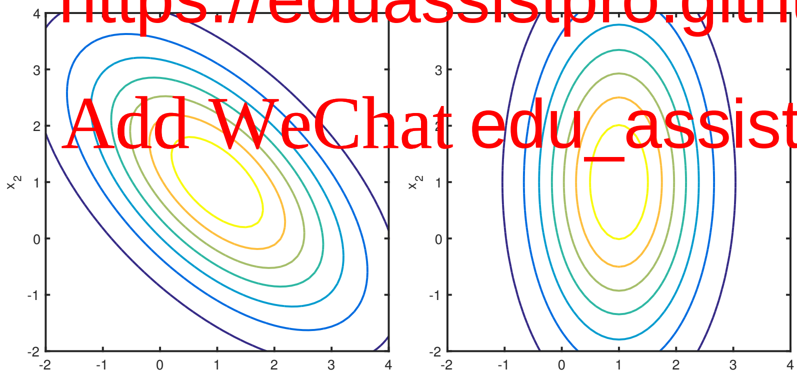
Figure 6: Fraction of residual variance from projectio
dimension

Covariance of data upon PCA mapping

- Covariance of data upon PCA mapping:

$$\frac{1}{n}\sum_{i=1}^{n}\varphi(x_i)\varphi(x_i)^\top = \frac{1}{n}\sum_{i=1}^{n}V_k^\top x_i x_i^\top V_k = V_k^\top \Sigma V_k = \Lambda_k$$

where Λ_k is diagonal matrix with $\lambda_1, \ldots, \lambda_k$ along diagonal.



PCA and linear regression

- Use k-dimensional PCA mapping $\varphi(x) = V_k^\top x$ with ordinary least squares
- (Assume rank of A is at least k, so $A^\top A$ has $\lambda_k > 0$)
- Data matrix is

$$= \qquad\qquad = \qquad {}_k \in \mathbb{R}^{n\times k}$$

$n \qquad\qquad n \quad k$

- Therefore, OLS solution is

$$\begin{aligned}\hat{\beta} &= (V_k^\top A^\top A V_k \\ &= \Lambda_k^{-1} V_k^\top A^\top b\end{aligned}$$

(Note: here $\hat{\beta} \in \mathbb{R}^k$.)

- Use $\hat{\beta} = \Lambda_k^{-1} V_k^\top A^\top b$ to predict on new $x \in \mathbb{R}^d$:

$$\begin{aligned} \varphi(x)^\top \hat{\beta} &= (V_k^\top x)^\top \Lambda_k^{-1} V_k^\top A^\top b \\ &= x^\top (V_k \Lambda_k^{-1} V_k^\top)(A^\top b) \end{aligned}$$

- ather

$$\hat{w} := (V_k \Lambda_k^{-1} V^\top \quad {}^\top$$

- This is called *principal component re* k is hyperparameter)

- Alternative hyper-parameterization: $\lambda > 0$; same as before but using the largest k such that $\lambda_k \geq \lambda$.

Spectral regularization

- PCR and ridge regression are examples of *spectral regularization*.
- For a function $g \colon \mathbb{R} \to \mathbb{R}$, write $g(M)$ to mean

$$d$$

  - I.e., g is applied to eigenvalues of
  - Generalizes effect of polynomials: e.g.,

$$M^2 = (V\Lambda V^{\mathsf{T}})(V$$

- **Claim**: Can write each of PCR and ridge regression as

$$\hat{w} = g(A^{\mathsf{T}}A)A^{\mathsf{T}}b$$

for appropriate function g (depending on λ).

- $\hat{w} = g(A^{\mathsf{T}}A)A^{\mathsf{T}}b$
- Ridge regression (with parameter λ): $g(z) = \frac{1}{z+\lambda}$
- PCR (with parameter λ): $g(z) = \mathbf{1}_{\{z \geq \lambda\}} \cdot \frac{1}{z}$
- Interpretation:

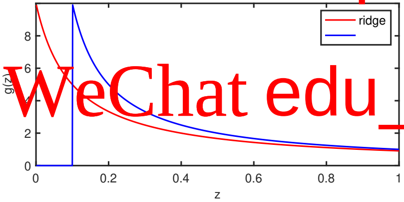


Figure 7: Comparison of ridge regression and PCR

- Let $A = \begin{bmatrix} \leftarrow & x_1^\top & \rightarrow \\ & \vdots & \\ \leftarrow & x_n^\top & \rightarrow \end{bmatrix} \in \mathbb{R}^{n \times d}$ (forget the $1/\sqrt{n}$ scaling)

- Try to approximate A with BC, where $B \quad \mathbb{R}^{n \times k}$ and

hich

uclidean

  - Think of B as the encodings of the dat
  - "Dimension reduction" when $k <$

- **Theorem** (Schmidt, 1907; Eckart-Y
solution is given by truncating the
singular value decomposition (SVD) of A

- Every matrix $A \in \mathbb{R}^{n \times d}$—say, with rank r—can be written as

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^{\mathsf{T}}$$

  - $v_1, \ldots, v_r \in \mathbb{R}^d$ (orthonormal *ri*
- Can also write as

$$A = US$$

where

  - $U = [u_1 | \cdots | u_r] \in \mathbb{R}^{n \times r}$, satisfies $U^{\mathsf{T}} U = I$
  - $S = \mathrm{diag}(\sigma_1, \ldots, \sigma_r) \in \mathbb{R}^{r \times r}$
  - $V = [v_1 | \cdots | v_r] \in \mathbb{R}^{d \times r}$, satisfies $V^{\mathsf{T}} V = I$

- Let A have SVD $A = \sum_{i=1}^{r} \sigma_i u_i v_i^\mathsf{T}$ (rank of A is r)
- Truncate at rank k (for any $k \leq r$): *rank-k SVD*

$$A_k := \sum_{i=1}^{k} \sigma_i u_i v_i^\mathsf{T}$$

-

  k k

  - $S_k = \operatorname{diag}(\sigma_1, \ldots, \sigma_k) \in \mathbb{R}^{k \times k}$
  - $V_k = [v_1 | \cdots | v_k] \in \mathbb{R}^{d \times r}$, satisfie

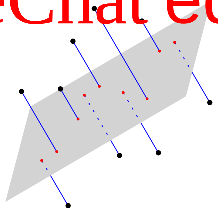
- **Theorem** (Schmidt/Eckart-Youn

$$\|A - A_k\|_F^2 = \min_{M:\operatorname{rank}(M)=k} \|A - M\|_F^2 = \sum_{i=k+1}^{r} \sigma_i^2$$

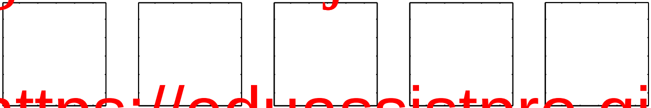
- Encoder: $x \mapsto \varphi(x) = V_k^{\mathsf{T}} x \in \mathbb{R}^k$
  - Encoding rows of A: $AV_k = U_k S_k$
- Decoder: $z \mapsto V_k z \in \mathbb{R}^d$
  - Decoding rows of $U_k S_k$: $U_k S_k V_k^{\mathsf{T}} = A_k$
- Same as k-dimensional PCA mapping!

r

squared distances to data points.



- Example: OCR data, compare original image to decoding of k-dimensional PCA encoding ($k \in \{1, 10, 50, 200\}$)



- Start with n documents, represent using "bag-of-words" count vectors
- Arrange in matrix $A \in \mathbb{R}^{n \times d}$, where d is vocabulary size

	aardvark	abacus	abalone
$\vdots$	$\vdots$	$\vdots$	

Application: Topic modeling (2)

- Rank k SVD provides an approximate factorization

$$A \approx BC$$

  where B $\mathbb{R}^{n\times k}$ and C $\mathbb{R}^{k\times d}$

-
-
-

- If rows of C were probability distrib

  $C_{t,w}$ as probability that word w a

- Start with ratings of movies given by users
- Arrange in a matrix $A \in \mathbb{R}^{n \times d}$, where $A_{i,j}$ is rating given by user i for movie j.
  - Netflix: $n = 480000$, $d = 18000$; on average, each user rates

-

$$B = \begin{bmatrix} \leftarrow & b_1^{\mathsf{T}} & \rightarrow \\ & \vdots & \\ \leftarrow & b_n^{\mathsf{T}} & \rightarrow \end{bmatrix} \in \mathbb{R}^{n \times k}, \quad C \quad {}^{k \times d}$$

with goal of minimizing $\|A - BC\|_F^2$

- Note: If all entries of A were observed, we could do this with truncated SVD.

- Need to find a low-rank approximation without all of A: *(low-rank) matrix completion*
  - Lots of ways to do this
  - Popular way (used in Netflix competition): based on "stochastic gradient descent" (discussed later)

Feature representations from matrix completion

- MovieLens data set ($n = 6040$ users, $d = 3952$ movies, $|\Omega| = 800000$ ratings)
- Fit B and C by using a standard matrix completion method (based on SGD, discussed later)

- k

- Some nearest-neighbor pairs (c_j,
  - Toy Story (1995), Toy Story 2 (1999)
  - Sense and Sensibility (1995), Emma (1
  - Heat (1995), Carlito's Way (1993)
  - The Crow (1994), Blade (1998)
  - Forrest Gump (1994), Dances with Wolves (1990)
  - Mrs. Doubtfire (1993), The Bodyguard (1992)