Machine Learning Principles

Class4 : Sept. 15

Linear Regression I

Instructor: Diana Kim

Today's Lecture

- **Linear Regression**

(1) preliminary: solving a linear equation using SVD & pseudo inverse

(2) **modeling**

- linear combination of basis functions (features) & parameters
- data preprocessing

(3) **learning**: Maximum Likelihood Estimation (MLE)

Mean Square Error Estimation (MMSE)

normal equation & its solution

- Preliminary
  - solving a linear equation using **SVD** & pseudo inverse

[1] Spectral Decomposition of Symmetric Matrix

A symmetric matrix (Σ) can be represented
by the matrix of eigenvectors (E) and eigenvalues (Λ).

$$\Sigma = \begin{bmatrix} & | & \\ e_1, e_2, & \ldots e_n \\ & | & \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ldots & 0 \\ \vdots & & & \\ 0 & \ldots & \ldots & \lambda_n \end{bmatrix} \cdot \begin{bmatrix} e_1^t \\ e_2^t \\ \ldots \\ e_n^t \end{bmatrix}$$

**convention: descending order

$$\Sigma = \lambda_1 \cdot e_1 e_1^t + \lambda_2 \cdot e_2 e_2^t + \ldots + \lambda_n \cdot e_n e_n^t$$

- e_i: eigenvectors, ${e_i}^t \cdot e_j = 0$ and $||e_i|| = 1$ (orthonormal)
- λ_i: eigenvalues (# non zeros = #rank)

[2] Singular Vector Decomposition of a **Rectangular** Matrix (A whose $n > m$)

$$A(n \times m) = \begin{bmatrix} & | & \\ u_1, u_2, ...u_n & & \\ & | & \end{bmatrix} \cdot \begin{bmatrix} \sqrt{\lambda_1} & 0 & ... & 0... \\ 0 & \sqrt{\lambda_2} & ... & 0...0 \\ \vdots & & & \\ 0 & ... & ... & \sqrt{\lambda_m} \\ \vdots & & & \\ 0 & ... & ... & 0 \\ 0 & ... & ... & 0 \end{bmatrix} \cdot \begin{bmatrix} v_1^t \\ v_2^t \\ ... \\ v_m^t \end{bmatrix}$$

- u_i: eigenvectors of $AA^t = E\Lambda E^t$ (n x n)
- v_i: eigenvectors of $A^tA = V\Lambda' V^t$ (m x m)
- λ_i: eigenvalues of both AA^t and A^tA

[3] Singular Vector Decomposition of a Rectangular Matrix (AA^t and A^tA)

when A is a ($n \times m, n > m$) matrix, both AA^t and A^tA are symmetric.

$$AA^t = \begin{bmatrix} \\ u_1, u_2, \ldots u_n \\ \\ \end{bmatrix} \cdot \underbrace{\begin{bmatrix} \lambda_1 & 0 & \ldots & 0 & \ldots & 0 & 0 \\ 0 & \lambda_2 & \ldots & 0 & \ldots & 0 & 0 \\ \vdots & 0 & \ldots & \lambda_m & & & \\ 0 & \ldots & 0 & \ldots & \ldots & 0 & 0 \\ 0 & \ldots & 0 & \ldots & \ldots & 0 & 0 \end{bmatrix}}_{[n \times n]} \cdot \begin{bmatrix} u_1^t \\ u_2^t \\ \ldots \\ u_n^t \end{bmatrix}$$

filling up (n-m) zeros!

$$A^tA = \begin{bmatrix} \\ v_1, v_2, \ldots v_m \\ \\ \end{bmatrix} \cdot \underbrace{\begin{bmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ldots & 0 \\ \vdots & & & \\ 0 & \ldots & \ldots & \lambda_m \end{bmatrix}}_{[\mathrm{m} \times m]} \cdot \begin{bmatrix} v_1^t \\ v_2^t \\ \ldots \\ v_m^t \end{bmatrix}$$

- Solving a Linear Equation
  $Ax = b$ (A is $n \times m$ matrix.)

[1] Solving a Linear Equation (two possible cases,)

$$Ax = b$$

- when b is in the column space of A
- when b is not in the column space of A

[2] Solving a Linear Equation: a general form

$$Ax = b \text{ (original)}$$

$$\downarrow$$

$$A^t Ax = A^t b \text{ (projection to the column space)}$$

[3] Solving a Linear Equation: exact vs. approximated solutions

$$Ax = b \text{ (original)}$$

$$A^t A x = A^t b \text{ (projection to the column space)}$$

- when b is in the column space of A:
  exact solutions
- when b is not in the column space of A:
  approximated solutions

[4] Solving a Linear Equation: (singular vs. non-singular)

$$Ax = b \text{ (original)}$$

$$A^t A x = A^t b \text{ (projection to the column space)}$$

- $A^t A$ is non − singular ($|A^t A| \neq 0$): one/unique solution. (invertible)
- $A^t A$ is singular ($|A^t A| = 0$) : infinite many solutions to $A^t b$ (invertible)

[5] Solving a Linear Equation: (example of singular A^tA)

$$A = \begin{bmatrix} 1 & 2 & 1 & 0 \\ 2 & 4 & 0 & 1 \end{bmatrix}$$

$$A^tA = \begin{bmatrix} 5 & 10 & 1 & 2 \\ 10 & 20 & 2 & 4 \\ 1 & 2 & 1 & 0 \\ 2 & 4 & 0 & 1 \end{bmatrix}$$

- How many independent column vector in A^tA ?
- How many possible solutions A^tA x = A^tb?
- How could we decide one?

[6] Solving a Linear Equation: non-singular case

$$Ax = b \text{ (original)}$$

$$A^t Ax = A^t b \text{ (projection to the column space)}$$

- $A^t A$ is non $-$ singular $(|A^t A| \neq 0)$: unique solution.

[7] Solving a Linear Equation: non-singular case

$$Ax = b \text{ (original)}$$

$$A^tAx = A^tb \text{ (projection to the column space)}$$

- A^tA is non − singular $(|A^tA| \neq 0)$: unique solution.
- $A^tAx = A^tb$

  $V\Lambda V^t x = V\ \Lambda'^{1/2}\ E^t b$

  $\Lambda V^t x = V^t V\ \Lambda'^{1/2}\ E^t b = \boldsymbol{\Lambda'^{1/2}\ E^t b}$

  $x = \boldsymbol{V\Lambda'^{-1/2}\ E^t b}$ **(what is the meaning of this)**

**what if A^tA is singular?

[8] Solving Linear Equation: singular case

$$Ax = b \text{ (original)}$$

$$A^t A x = A^t b \text{ (projection to the column space)}$$

- $A^t A$ is singular ($|A^t A| = 0$) : infinite many solutions to $A^t b$

$$V \Lambda V^t x = A^t b$$
$$x = (V \Lambda V^t)^\dagger A^t b$$
$$= V {\Lambda^*}^{-1/2} E^t b$$

[9] Solving Linear Equation

$$Ax = b \text{ (original)}$$

$$A^tAx = A^tb \text{ (projection to the column space)}$$

- A^tA is singular ($|A^tA| = 0$) : infinite many solutions to A^tb

$$x = \boxed{V{\Lambda^*}^{-1/2}E^t}b$$

Pseudo Inverse of $A^\dagger$

- Pseudo-Inverse (using SVD)

  Generalization of the notion of inverse matrix.

[1] Singular Vector Decomposition of a Rectangular Matrix (Pseudo-Inverse)

$$\mathrm{D} = \underset{\mathrm{U}}{\begin{bmatrix} \Big| \\ u_1, u_2, \ldots u_n \\ \Big| \end{bmatrix}} \cdot \begin{bmatrix} \sqrt{\lambda_1} & 0 & \ldots & 0 \\ 0 & \sqrt{\lambda_2} & \ldots & 0 \\ \vdots & & & \\ 0 & & \sqrt{\lambda_{m-1}} & 0 \\ 0 & \ldots & \ldots & 0 \\ 0 & \ldots & \ldots & 0 \\ 0 & \ldots & \ldots & 0 \end{bmatrix} \cdot \underset{V^t}{\begin{bmatrix} v_1^t \\ v_2^t \\ \ldots \\ v_m^t \end{bmatrix}}$$

$$\mathrm{D}^{\dagger} = \underset{\mathrm{V}}{\begin{bmatrix} \Big| \\ v_1, v_2, \ldots v_m \\ \Big| \end{bmatrix}} \cdot \begin{bmatrix} \frac{1}{\sqrt{\lambda_1}} & 0 & \ldots & 0 & \ldots & 0 \\ 0 & \frac{1}{\sqrt{\lambda_2}} & \ldots & 0 & \ldots & 0 \\ \vdots & & & & & \\ 0 & & \frac{1}{\sqrt{\lambda_{m-1}}} & 0 & \ldots & 0 \\ 0 & \ldots & \ldots & \boxed{0} & \ldots & 0 \end{bmatrix} \cdot \underset{U^t}{\begin{bmatrix} u_1^t \\ u_2^t \\ \ldots \\ u_n^t \end{bmatrix}}$$

pseudo-inverse

dropping the axes corresponding to the zero eigenvalues

[2] Singular Vector Decomposition of a Rectangular Matrix (Pseudo-Inverse)

ex) compute the pseudo-inverse of the matrix below. [rectangular matrix]

$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

[3] Singular Vector Decomposition of a Rectangular Matrix (Linear System: D$x = b$)

$Dx = b$

- no solution

$D^t D x = D^t b$

- projection to column space (approximated)
- exist solution (one / infinite many solution)

$(D^t D)^\dagger (D^t D)x = (D^t D)^\dagger D^t b$

x $= (D^t D)^\dagger D^t b$

- by using pseudo-inverse,
  find a solution in the subspace

- Linear Regression Problem

[1] What is the regression problem?

- Learning the function f
  to predict continuous y
  given the value of M dimensional input data $(x_1, x_{2,}, \ldots . x_{m,})$

$$y = f\,(x_1, x_{2,}, \ldots . x_{m.})$$

(functional relation between x and y)

[2] What is the regression problem? (two kinds of LR)

- each feature dimension has semantic
  "prediction of house market price"
  single house/ townhome, square feet, garden size, public school scores

- no semantic, a whole vector represents an image/ audio /texts

  this problem will be our focus in this course!

[3] What is the regression problem? (semantic regression example)

[A car's fuel efficiency in miles per gallon]

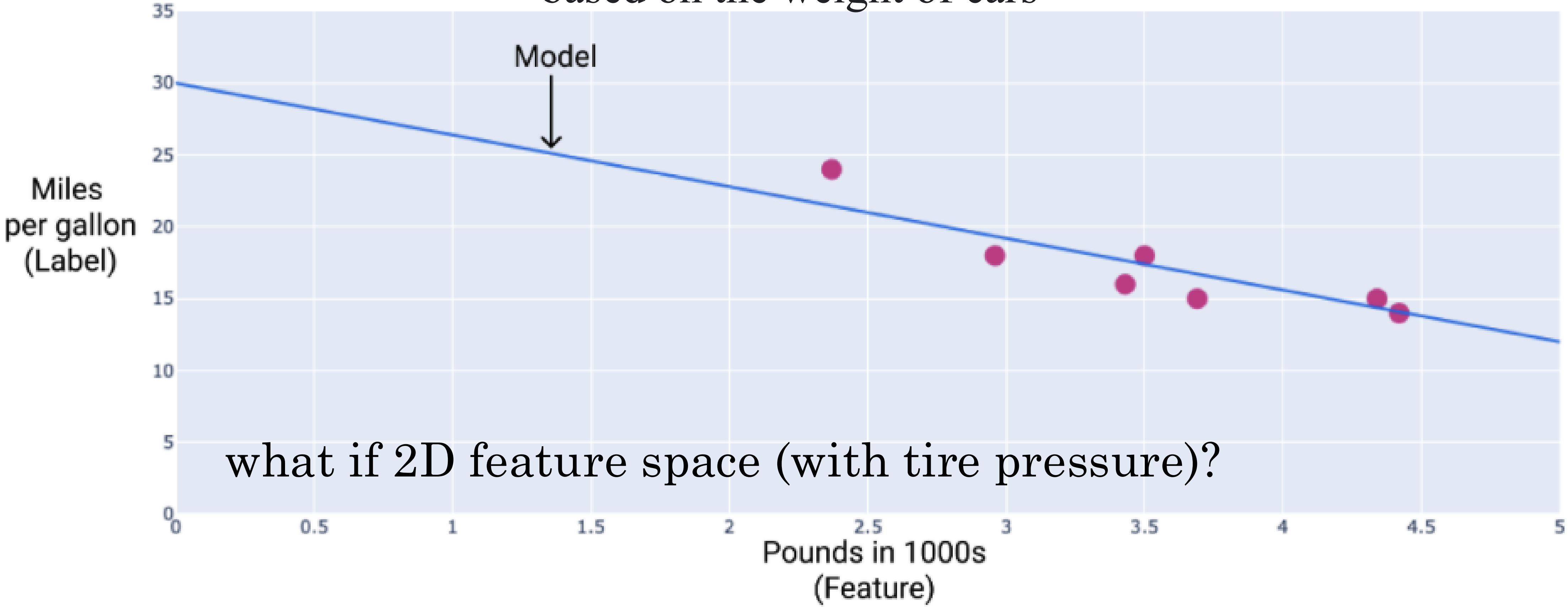
Pounds in 1000s (features)	Miles Per Gallon (Label)
3.5	18
3.69	15
3.44	18
3.43	16
4.34	15
4.42	14
2.37	24

From https://developers.google.com/machine-learning/crash-course/linear-regression

[4] What is the regression problem? (semantic regression example)

Regression Model: prediction of Miles/ Gallon based on the weight of cars

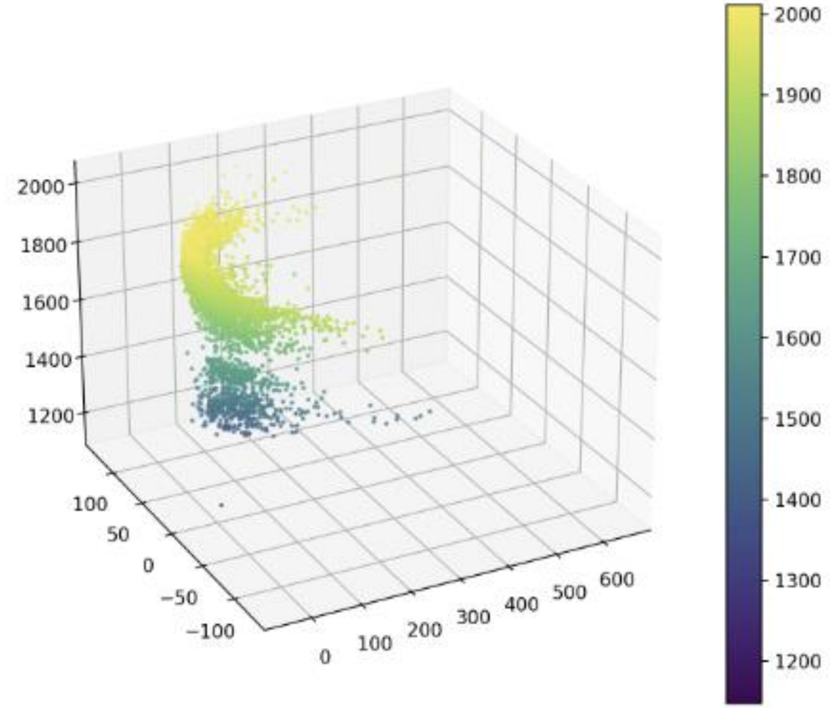
- $y = f(x)$



what if 2D feature space (with tire pressure)?

From https://developers.google.com/machine-learning/crash-course/linear-regression

[5] What is the regression problem? (non-semantic regression example)



(a) the most accurate image: id 9593: (gt: 1925 vs. prediction 1924.89)

(b) the least accurate image: id 847: (gt: 1455 vs. prediction 1885)

- $y = f(x_1, x_2)$
- prediction of year of made of fine art paintings based on the 2-d PCA space of hidden embeddings from a Deep-CNN style classifier

[6] What is the regression problem? (many different names)

$$Y=\beta_0+\beta_1 x_1+\varepsilon_0$$

Dependent Variable	Independent Variable
Regressand	Regressor
Response	Covariate
Endogenous	Exogenous
Target	Feature
Predicted	Predictor
Explained Variable	Explanatory Variable

[7] What is the regression problem? (scalar domain function examples)

$$y = ax + b$$ [linear]

$$y = ax^3 + bx^2 + c$$ [non-linear]

$$y = a \exp\{-\frac{(x-\mu_1)^2}{2\sigma_1}\} + b \exp\{-\frac{(x-\mu_2)^2}{2\sigma_2}\} + c \exp\{-\frac{(x-\mu_3)^2}{2\sigma_3}\}$$[non-linear]

[7] What is the linear regression problem? (linear representation)

$$y = ax + b \quad \longleftrightarrow \quad y = \begin{bmatrix} x & 1 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix}$$

$$y = ax^3 + bx^2 + c \quad \longleftrightarrow \quad y = \begin{bmatrix} x^3 & x^2 & 1 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

$$y = a\exp\{-\frac{(x-\mu_1)^2}{2\sigma_1}\} + b\exp\{-\frac{(x-\mu_2)^2}{2\sigma_2}\} + c\exp\{-\frac{(x-\mu_3)^2}{2\sigma_3}\}$$

Regression modeling can be expressed
as a linear combination of parameters and data features, hence the name is **Linear Regression.**

[8] What is the linear regression problem? (feature engineering)

effective feature engineering is the key to linear regression problem.

- choice of basis functions: nature of target tasks
- <u># (num) features: complexity</u>
- no collinearity *

[9] What is the linear regression problem? (intuitive way of learning)

$$y = ax + b$$

$$y = ax^3 + bx^2 + c$$

$$y = a \exp\{-\frac{(x-\mu_1)^2}{2\sigma_1}\} + b \exp\{-\frac{(x-\mu_2)^2}{2\sigma_2}\} + c \exp\{-\frac{(x-\mu_3)^2}{2\sigma_3}\}$$

Q: how could we learn the a, b, c?

[10] What is the linear regression problem? (intuitive way of learning)

(0) we have a data: a set of N samples: $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_N, y_N)\}$
(1) set a hypothetical space based on data available
- choice of basis functions: the nature of target tasks
- # (num) features (complexity): available number of data samples
- no collinearity

[11] What is the linear regression problem? (intuitive way of learning)

(0) we have a data: a set of samples: $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_N, y_N)\}$

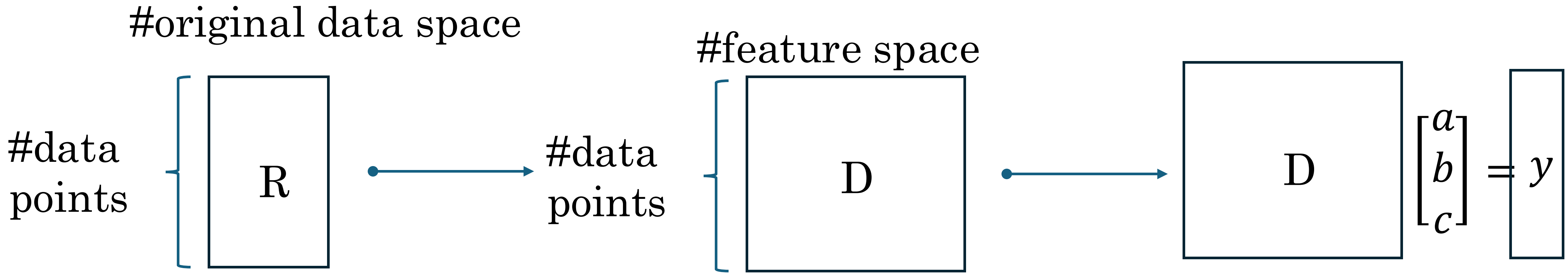
(1) set a hypothetical space based on data available

- choice of basis functions: nature of target tasks
- # (num) features (complexity): available number of data samples
- no collinearity

(2) apply the data points to the model

(3) define and solve a linear system

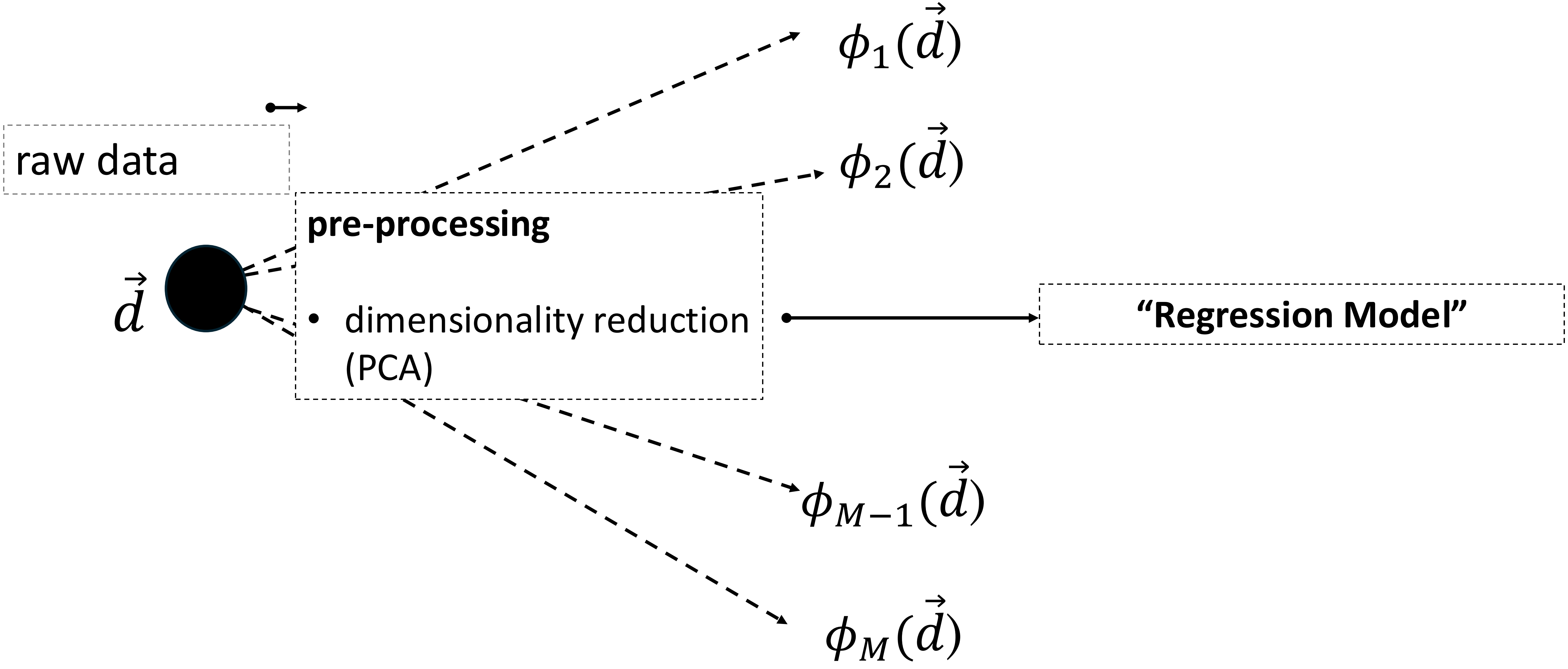
[Linear Regression Process]



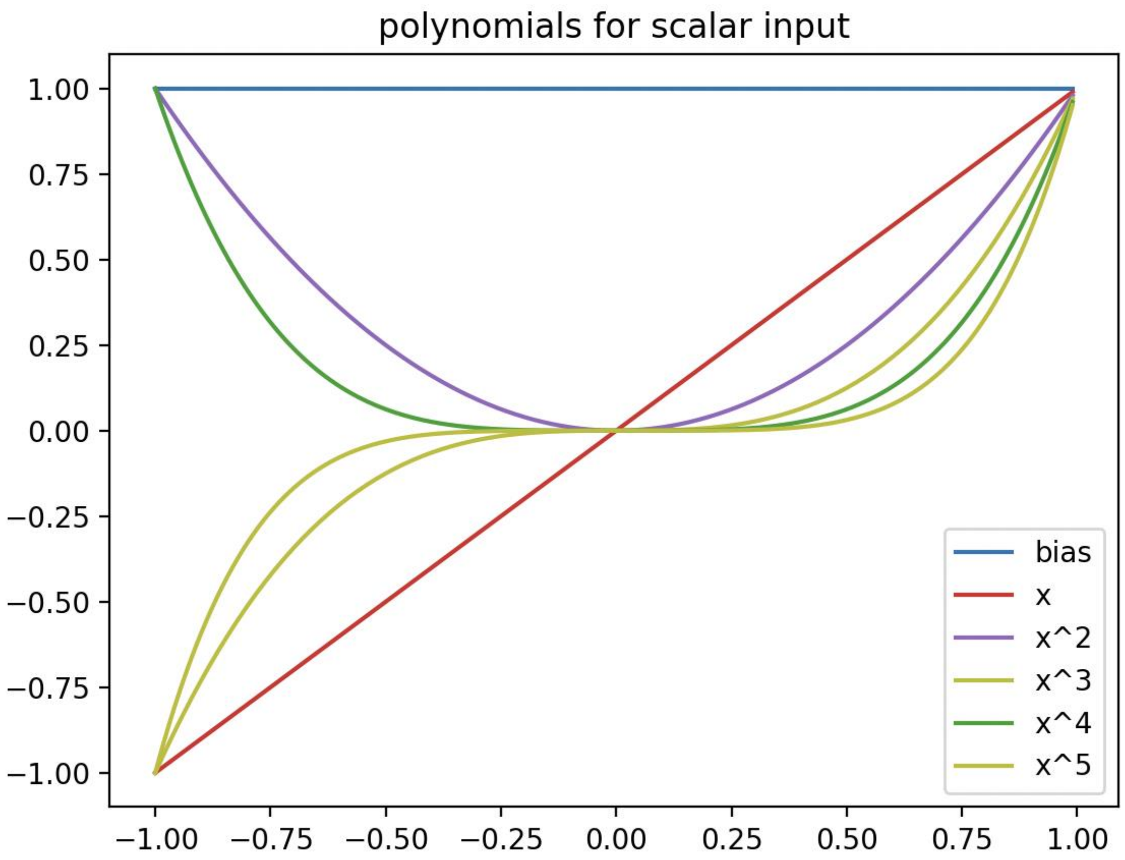
▪ Basis Functions (Feature Functions)

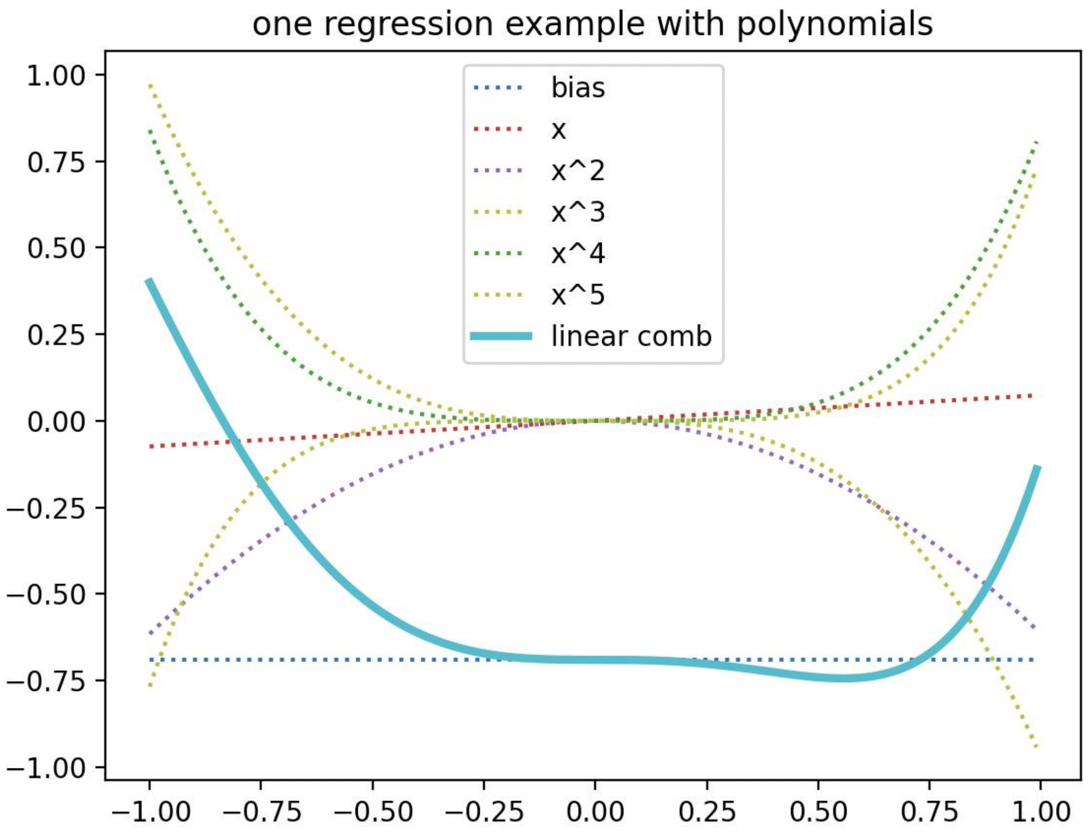
-a set of functions sharing the raw data as an input
-elementary functions to describe a function we target

[1] Basis Functions (a set of functions on the space of raw data)



[2] Basis Functions (polynomial expansion: scalar)





- This example shows the case when data input is scalar.
- Q: what if input is a 2D data vector? How would you draw the plot?

[3] Basis Functions (polynomial expansion:2d)

[**quadratic** polynomial example for 2-d raw data]

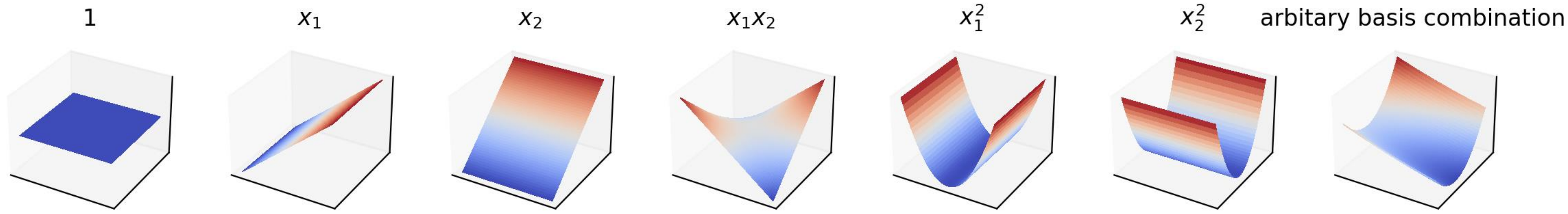
$$\vec{x} = (x_1, x_2)$$

$$\left(1, x_1, x_2, x_1x_2, x_1^2, x_2^2\right)$$

	1	x_2	${x_2}^2$
1	1	x_2	${x_2}^2$
x_1	x_1	x_1x_2	$\times$
${x_1}^2$	${x_1}^2$	$\times$	$\times$

[4] Basis Functions (polynomial expansion: 2d)

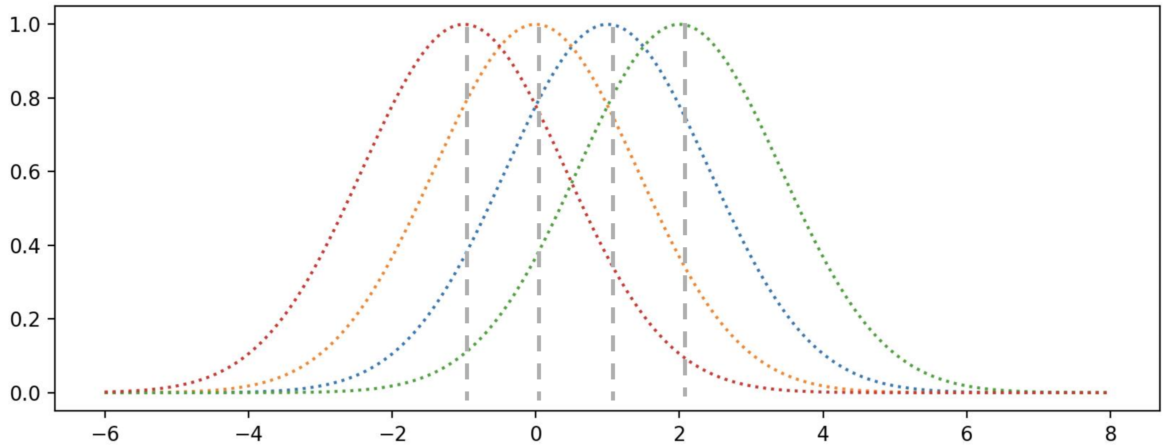
[six quadratic polynomial basis functions for 2-d raw data]



- linear combination will form a curved plane!

[5] Basis Functions (Gaussian basis function/ Radial basis function)

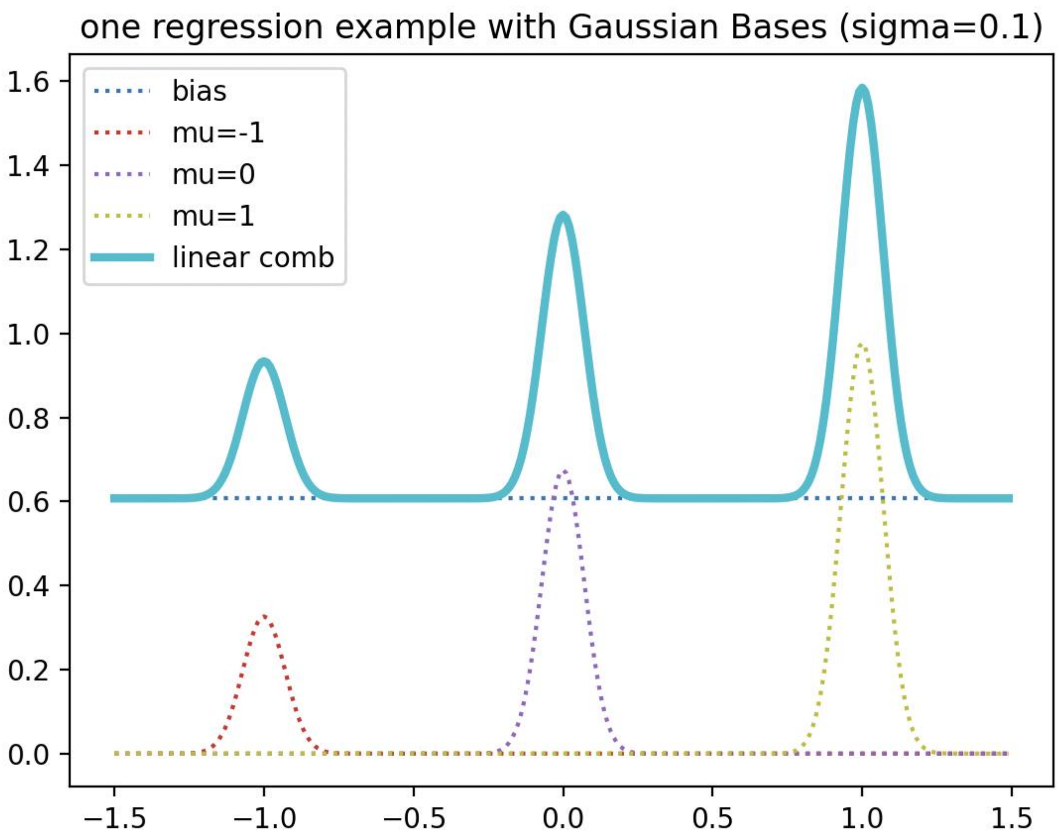
$$\phi_j = \exp\{-\frac{(x-\mu_j)^2}{2\sigma^2}\}$$

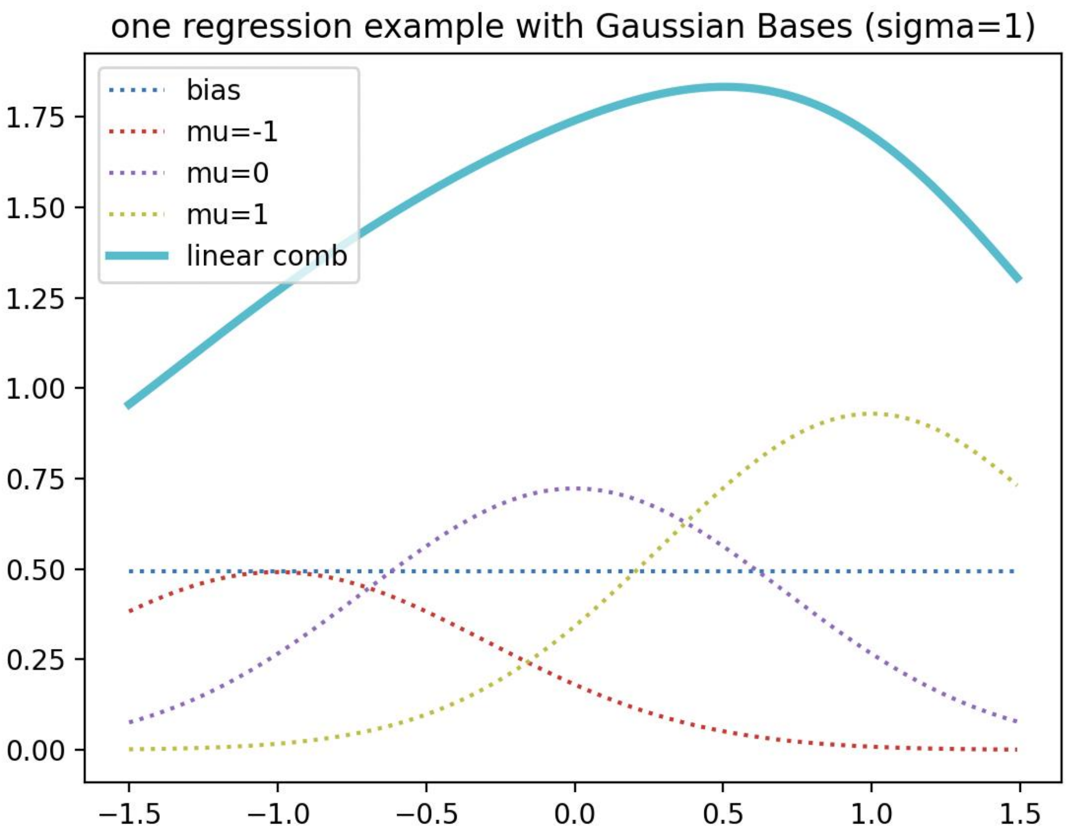


(-1 0 1 2)

Q: the locations of μ_j? (dense / sparse)
Q: the magnitude of σ^2?
(small: local and spiky vs. large: global and smooth)

[6] Basis Functions (Gaussian basis function)

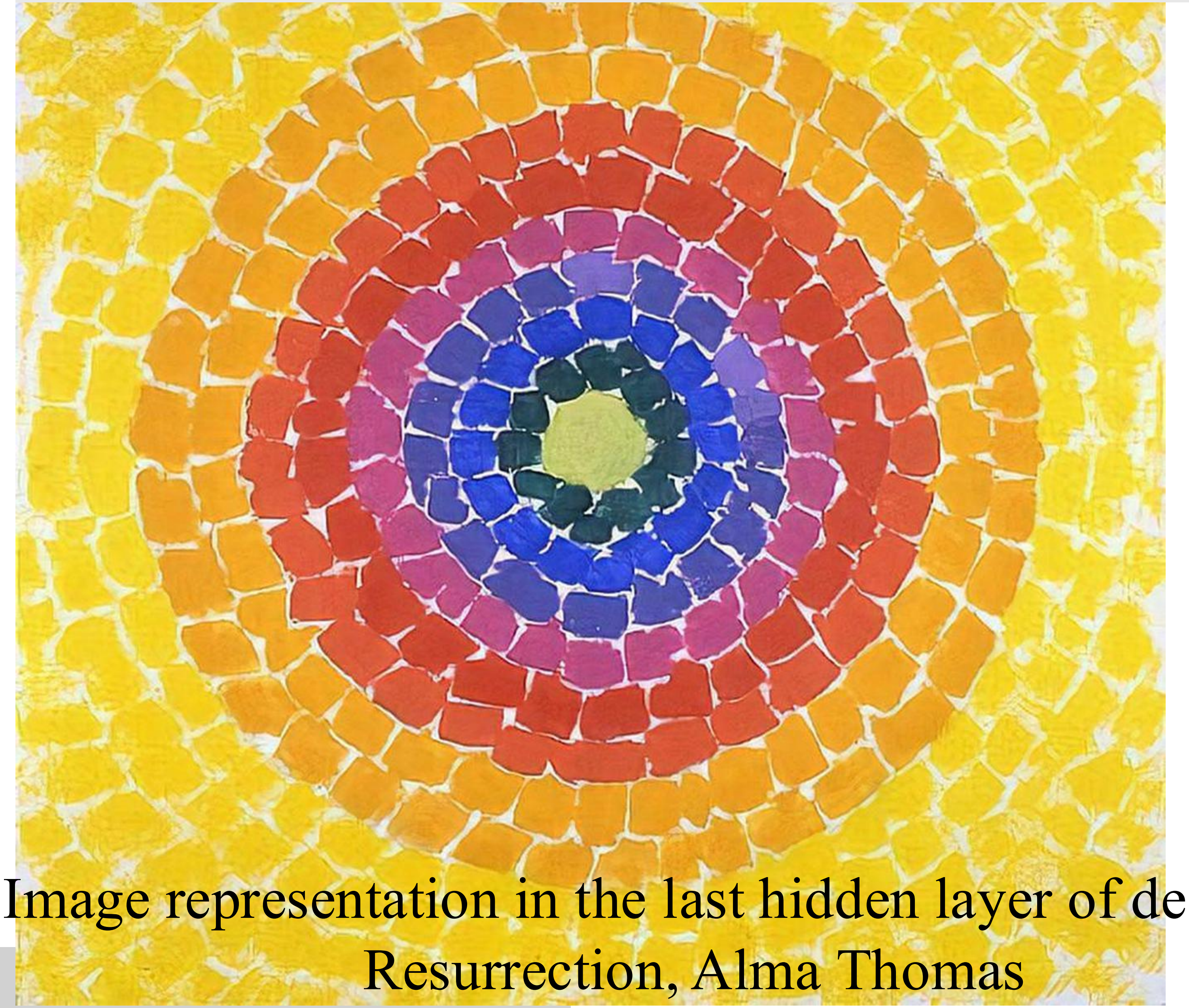




The magnitude of sigma determines the influence of the Gaussian over other neighboring areas.

- Data Preprocessing

[1] Data Preprocessing (a data vector in a very high dimensional space)

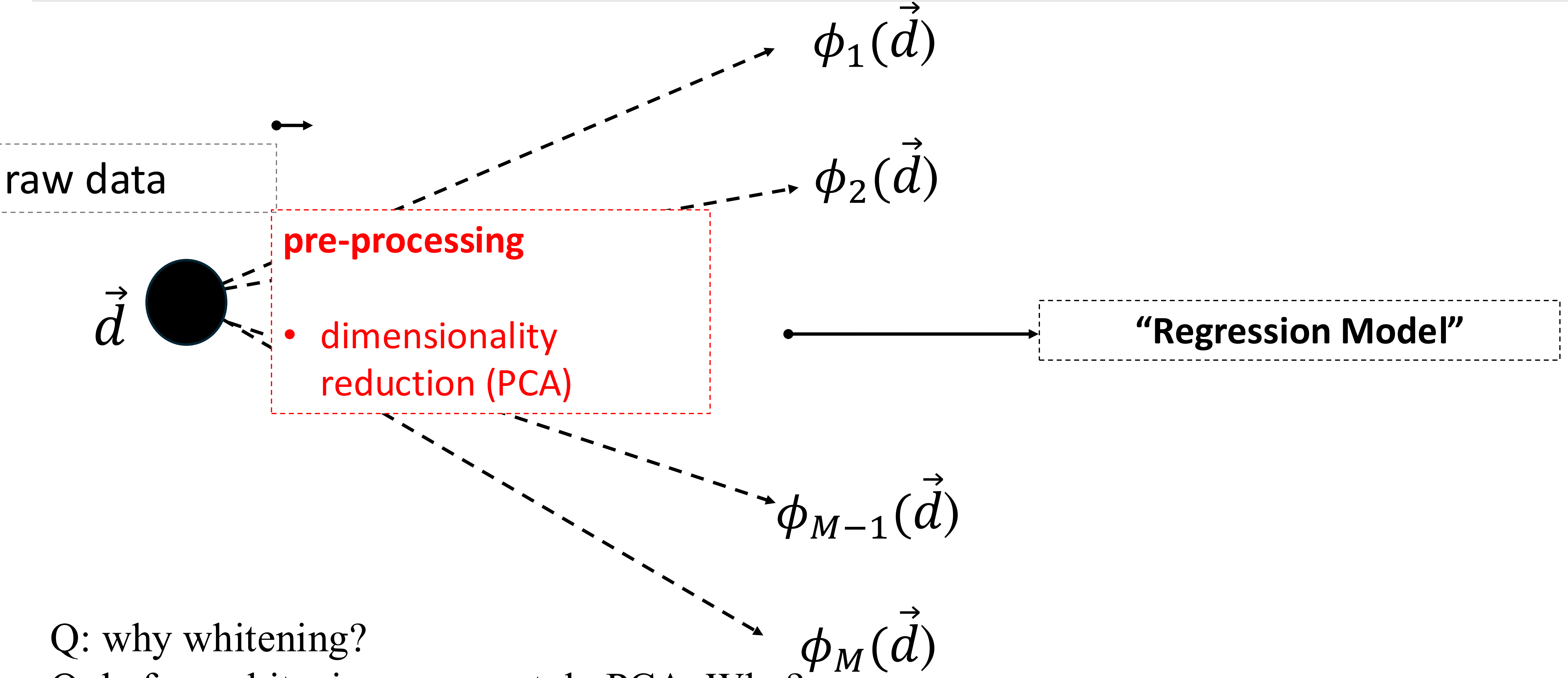


2.96458267e-02	1.08750183e-02	1.02128655e-01	4.72795311e-03
3.65977734e-02	2.39002742e-02	1.02527821e-02	2.37334445e-02
1.44995004e-02	2.45263092e-02	4.37902249e-02	2.01957620e-04
1.88012738e-02	3.28723639e-02	2.84272023e-02	9.55437776e-03
4.14613076e-02	1.39710018e-02	1.36978943e-02	1.87548213e-02
8.24719146e-02	4.43845317e-02	3.53335054e-03	7.63716456e-03
9.22968891e-03	3.64266671e-02	2.88719591e-02	1.30119538e-02
2.02936288e-02	1.33205568e-02	1.15464339e-02	8.20994973e-02
1.77106280e-02	8.52256175e-03	1.17111830e-02	1.45943370e-02
9.04859081e-02	4.89737056e-02	4.16163765e-02	1.04203597e-02
2.58007385e-02	4.69408296e-02	4.55647372e-02	1.24655450e-02
2.41902079e-02	2.42145211e-02	4.81210562e-04	8.90940428e-03
6.85443357e-02	4.78595048e-02	1.18027581e-02	1.17037995e-02
1.90981105e-02	1.00829145e-02	5.23220561e-03	3.84746492e-02
8.81355628e-02	3.36198583e-02	5.35092168e-02	4.87123579e-02
8.99140537e-03	5.39787523e-02	4.79393527e-02	2.99579669e-02
1.71675645e-02	3.76332477e-02	7.88647458e-02	3.79528590e-02
4.51750029e-03	9.38767791e-02	3.61216962e-02	1.98117495e-02
3.94294709e-02	1.14793226e-01	1.48017062e-02	6.40132884e-03
8.16167146e-03	6.94637448e-02	3.43800858e-02	8.04584008e-03
1.55022442e-01	2.59600277e-03	3.20520252e-02	1.00370266e-01
6.82575488e-03	3.21909995e-03	6.07831627e-02	5.22131985e-03
4.10482734e-02	5.29111736e-02	4.37461957e-02	4.37461808e-02
4.10865359e-02	9.59158130e-03	4.68185917e-02	7.04082549e-02
5.19240461e-03	9.47480425e-02	1.72703192e-02	1.32609099e-01
1.84857957e-02	2.34019849e-03	2.21508313e-02	3.19128227e-03
1.03731174e-02	7.90489465e-02	3.40001471e-02	2.08658073e-02
3.63909267e-03	2.93061193e-02	1.79619715e-02	3.92507110e-03
1.22312911e-01	4.27385271e-02	4.02529091e-02	6.87315594e-03
1.79619640e-02	1.44496362e-03	3.47868539e-04	2.03075245e-01
2.45202169e-01	1.26138151e-01	1.07999377e-01	1.46901429e-01
9.70007405e-02	1.03836969e-01	1.09804377e-01	1.04106106e-01
8.70869756e-02	8.81577432e-02	7.79228508e-02	9.59928930e-02
2.06121951e-01	2.38734394e-01	1.37491360e-01	7.11895898e-02
9.10348147e-02	1.08147562e-01	8.93435627e-02	8.45326930e-02
8.54639262e-02	7.94288218e-02	7.84831643e-02	6.98279142e-02
6.67123348e-02	7.02826679e-02	1.02719694e-01	1.04542613e-01
1.12103589e-01	8.02482218e-02	1.26211137e-01	1.22251317e-01
1.18328705e-01	9.65996012e-02	9.47735459e-02	8.21543038e-02
7.41177499e-02	1.03439212e-01	1.14290312e-01	1.15447372e-01
1.28355548e-01	1.06327742e-01	7.30694234e-02	6.20305464e-02
1.06132567e-01	7.94187784e-02	8.86070132e-02	8.47868249e-02
1.07920051e-01	8.36525112e-02	6.55624866e-02	7.80229717e-02
8.64467472e-02	8.55527893e-02	1.10759147e-01	1.32106051e-01
7.44441077e-02	5.27140088e-02	1.08958408e-01	8.06024447e-02
9.61078107e-02	9.56790447e-02	1.04670644e-01	8.01085979e-02
6.94930553e-02	7.93032944e-02	9.49410051e-02	7.71025643e-02
1.05781302e-01	1.46627113e-01	6.05126023e-02	4.13953587e-02
1.23981662e-01	1.08231083e-01	1.34232193e-01	1.18365087e-01
1.09341882e-01	8.85261148e-02	8.09772611e-02	8.30637813e-02
1.13967851e-01	8.66363198e-02	1.12511240e-01	1.40123367e-01
6.65690452e-02	4.43194956e-02	1.57082587e-01	1.04566351e-01
9.03969407e-02	1.14839226e-01	1.21048279e-01	9.68690664e-02
9.22555625e-02	1.10619672e-01	1.21558294e-01	8.79304186e-02
1.04587585e-01	1.42198473e-01	8.80973265e-02	4.42507043e-02

Image representation in the last hidden layer of deep-CNN

Resurrection, Alma Thomas

[2] Data Preprocessing



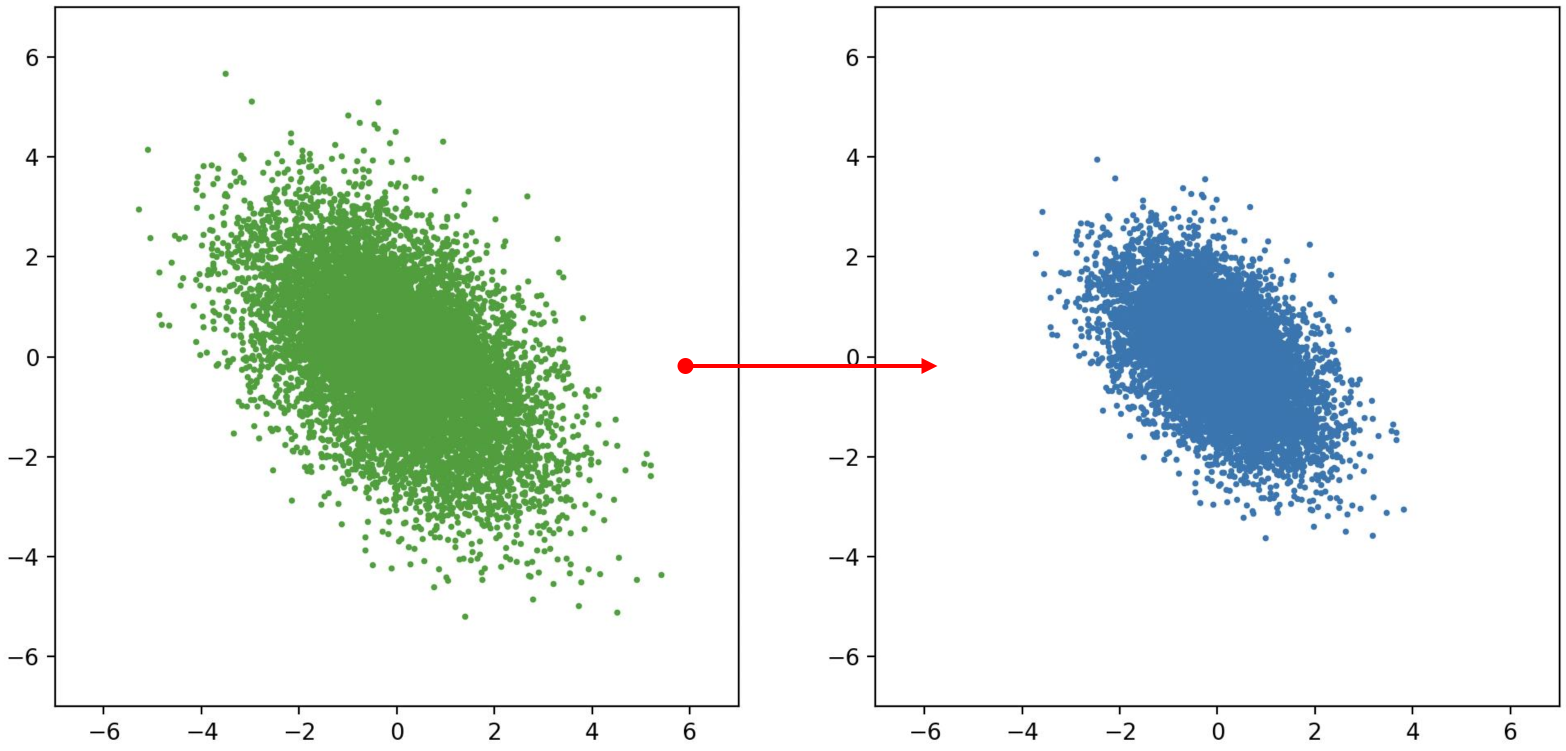
Q: why whitening?

Q: before whitening, we must do PCA. Why?

[3] Data Preprocessing (centering, normalization, standardization, whitening)

- Centering: $\vec{x} - E[\vec{X}]$
- Normalization: $\dfrac{\vec{x}}{||x||}$
- Standardization: $\dfrac{x_i - E[X_i]}{\sqrt{VAR[X_i]}}$ (element-wise operation) $\longleftarrow$ (no decorrelation)
- Whitening : $Y = AX + \vec{b}$ when $\mu_Y = \vec{0}$ and COV(Y) = I

[4] Data Preprocessing (standardization)



- standardization effect is not same as whitening.
- scaling but the scaling factor does not reflect eigenvalues./vectors

[5] Data Preprocessing (decorrelation by whitening / PCA)

Decorrelation is a useful step in regression modeling
to reduce a collinearity effect. Collinearity effect refer to the situation in RL
where two or more features are highly correlated. (redundancy)

[6] Data Preprocessing (decorrelation by whitening / PCA)

[collinearity example] $X_2 \approx 2X_1$ (hard to know existence of collinearity without the spectral analysis on covariance)

	X_1	X_2	X_3
#1	1.4	2.8	3.2
#2	2.2	4.4	3.3
#3	3.1	6	1.2
#4	1.7	3.4	0.2
#5	4.6	8	0.9
#6	2.2	4	2.7
#7	1.2	2.3	7.6
#8	0.3	0.8	3.2
#9	2.5	5	0.9
#10	1.8	3.5	1.1
…			

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} =$$

	Y
#1	1
#2	1.3
#3	0.8
#4	0.3
#5	1
#6	4
#7	3.6
#8	2.9
#9	2.5
#10	0
…	

- Hard to interpret the model

$$\begin{aligned} Y &= a\,X_1 + b\,X_2 + cX_3 \\ &= a\,X_1 + 2b\,X_1 + cX_3 \end{aligned}$$

- Unstable in learning

(very small eigenvalues / pseudo inverse very large)

- Linear Regression by MLE

  Learning by Minimum Mean Square Error (MMSE)

[0] Recall slide **: Learning, MLE vs. MAP

Frequentist vs. Bayes Estimation

- $w* = argmax\ P(D|w)$: **M**aximum **L**liklihood **E**stimation (MLE)
- $w* = argmax\ p(w|D) = \frac{p(D|w)p(w)}{p(D)}$: **M**aximum **A** **P**osteriori Estimation (**MAP**)

Frequentist assumes w (parameter) as fixed values and perform MLE to estimate the parameters. MLE can be interpreted as a special case of MAP when the prior density $p(w)$ is uniform.

[1] Linear Regression by MLE (Probabilistic Modeling)

$$y = f(x) + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2)$$

$$y = \overrightarrow{\Phi(x)}^t \cdot \vec{w} + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2)$$

[the vector of basis functions] [the vector of parameters]

ε errors from :

- imperfection hypothesis space (factors and basis functions)
- error from measurement

[2] Linear Regression by MLE

$$y = f(x) + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2)$$

$$y = \overrightarrow{\Phi(x)}^t \cdot \vec{w} + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2)$$

want to estimate $\vec{w}$!
when data samples $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$

- Q: the distribution of $p\left(\vec{y} \middle| \vec{w}, \vec{X}, \Phi\right)$?

[3] Linear Regression by MLE

$$w^* = argmax_w \; p(\vec{y}|\vec{w}, \vec{x}, \Phi)$$

$$= argmax_w \; \mathcal{N}_y(\overrightarrow{\Phi(x)} \cdot \vec{w}, \; \sigma^2 \mathrm{I})?$$

$$= argmin_w \; ||\vec{y} - \overrightarrow{\Phi(x)} \cdot \vec{w}||^2$$

MLE becomes
Minimum Mean Square Error Problem.

[4] Linear Regression by MLE (MMSE & Normal Equation)

$$w^* = argmax_w \, p(\vec{y} | \overrightarrow{w}, \overrightarrow{x}, \Phi)$$

$$= argmax_w \, \mathcal{N}_y(\overrightarrow{\Phi(x)} \cdot \vec{w}, \ \sigma^2 \mathrm{I})?$$

$$\boxed{= \arg\min_w ||\vec{y} - \Phi(\vec{x}) \cdot \vec{w}||^2}$$

MLE becomes
Minimum Mean Square Error Problem

$$J(\vec{w}) = ||\vec{y} - \Phi(\vec{x}) \cdot \vec{w}||^2$$

$$J(\vec{w}) = (\vec{y}^t - \vec{w}^t \cdot \Phi(\vec{x})^t) \cdot (\vec{y} - \Phi(\vec{x}) \cdot \vec{w})$$

$$\nabla J(\vec{w}) = -2 \cdot \Phi(\vec{x})^t \cdot (\vec{y} - \Phi(\vec{x}) \cdot \vec{w}) = 0$$

$$\boxed{\Phi(\vec{x})^t \cdot \Phi(\vec{x}) \cdot \vec{w} = \Phi(\vec{x})^t \cdot \vec{y}}$$

Normal Equation

[5] Linear Regression by MLE (Data Matrix $\overrightarrow{\Phi(x)}$)

data dimension: D

raw data

data: N

x_1
x_2
…
…
…
…
x_N

feature dimension: M

data matrix $\overrightarrow{\Phi(x)}$

$\phi_1(x_1)$	$\phi_2(x_1)$	…	$\phi_M(x_1)$
$\phi_1(x_2)$	$\phi_2(x_2)$	…	$\phi_M(x_2)$
…	…		
…	…		
…	…		
…	…		
$\phi_1(x_N)$	$\phi_2(x_N)$	…	$\phi_M(x_N)$

[6] Linear Regression by MLE (solving Normal Equation)

$$\Phi(\vec{x})^t \cdot \Phi(\vec{x}) \cdot \vec{w} = \Phi(\vec{x})^t \cdot \vec{y}$$

The three possible cases in solving the linear equation:

- invertible (Rank M)
- invertible (Rank M) but close to singular (very small eigenvalues)
- non – invertible (Rank $<M$)

[7] Linear Regression by MLE (solving Normal Equation)

$$\Phi(\vec{x}) \cdot \vec{w} = \vec{y}$$

- no solution (over determined equation)

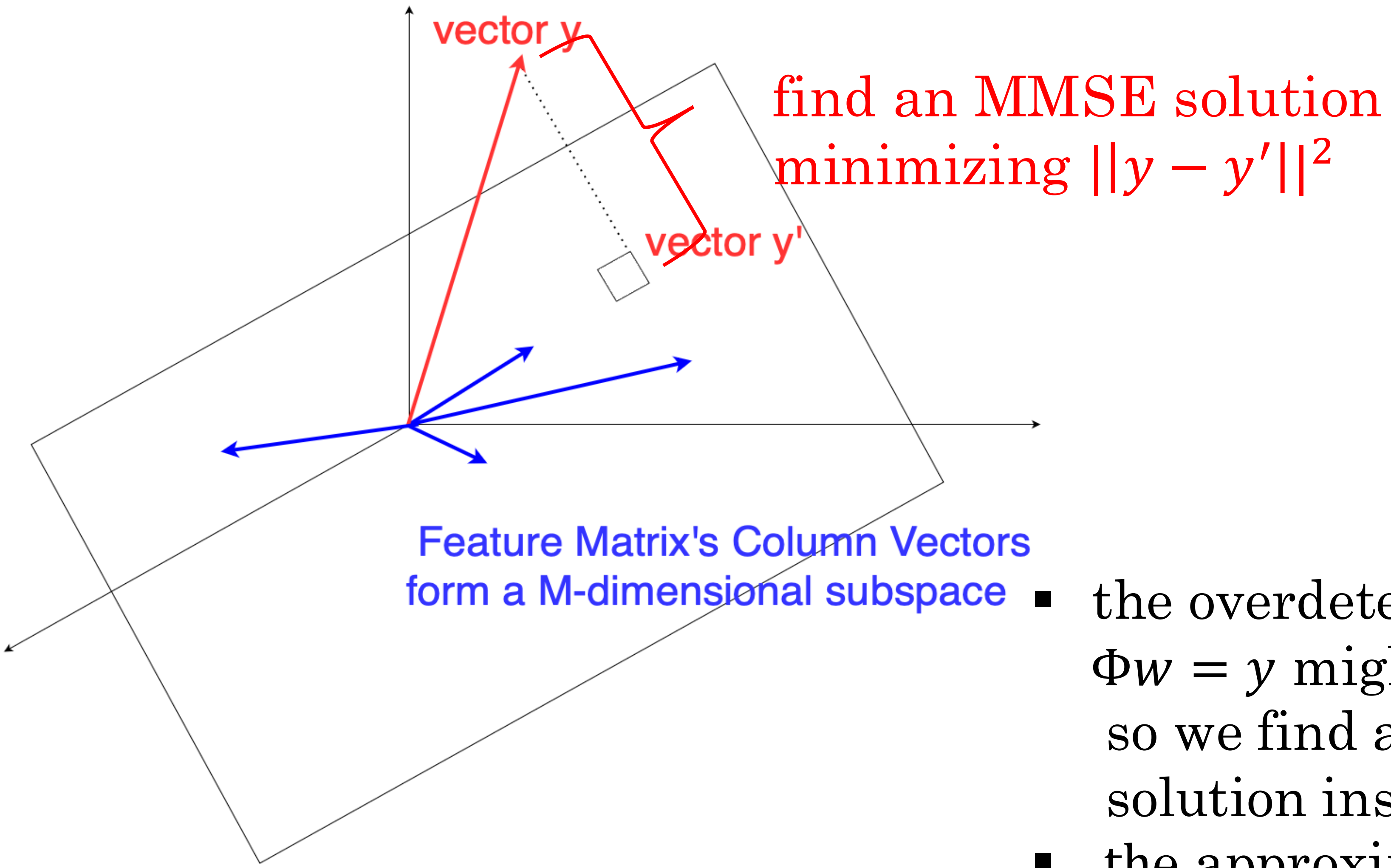
$$\Phi(\vec{x})^t \cdot \Phi(\vec{x}) \cdot \vec{w} = \Phi(\vec{x})^t \cdot \vec{y}$$

- projection to column space (approximated)
- exist solution (one / infinite many solution)

$$\vec{w} = (\Phi(\vec{x})^t \cdot \Phi(\vec{x}))^\dagger \cdot \Phi(\vec{x})^t \cdot \vec{y}$$

- by computing the pseudo-inverse, find a solution in the approximated space

[8] Linear Regression by MLE (Geometric Interpretation of MMSE)



- the overdetermined system $\Phi w = y$ might not have a solution, so we find an approximated solution instead. ($\Phi^t \Phi w = \Phi^t y$)
- the approximated solution is an MMSE.