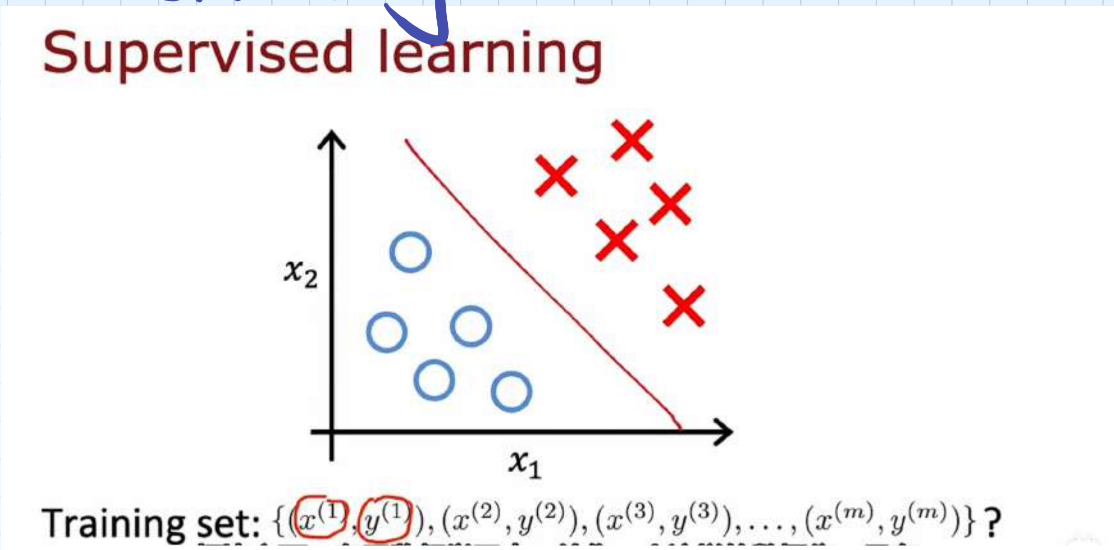
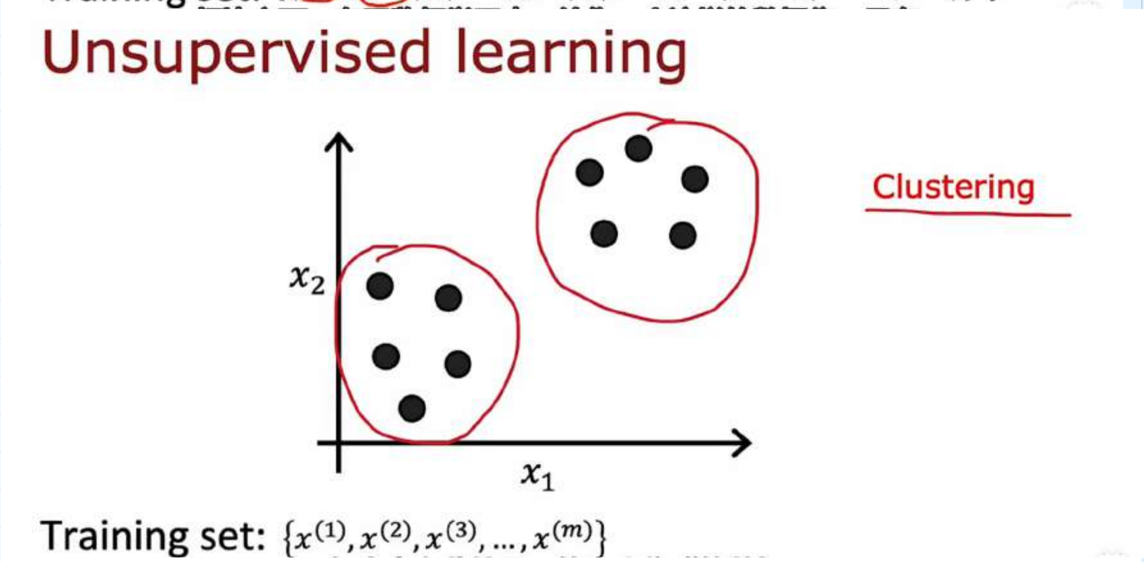
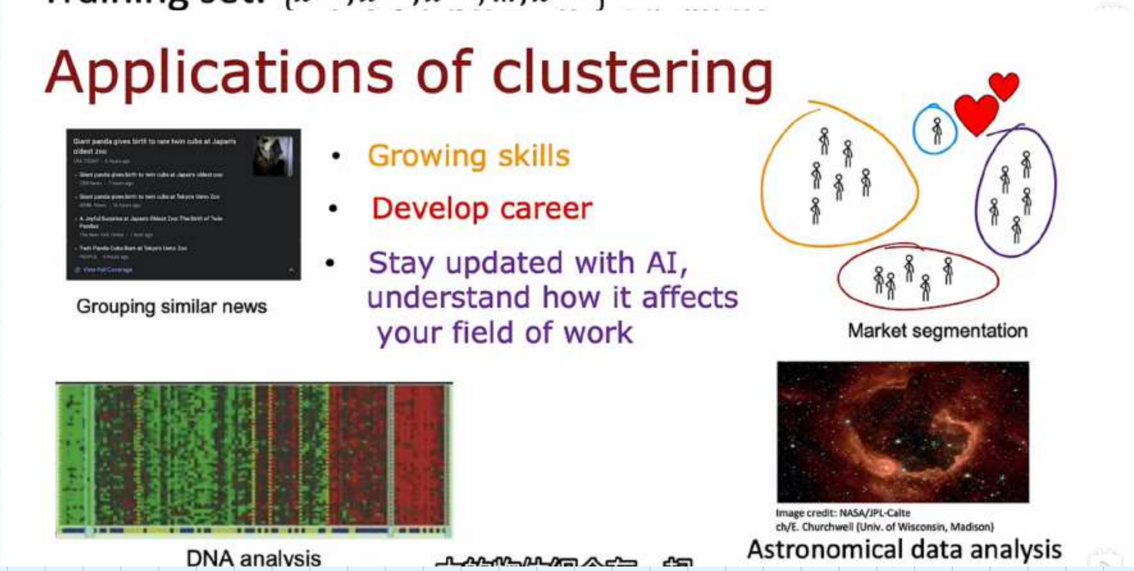
Unsupervised Learning

Clustering 聚类

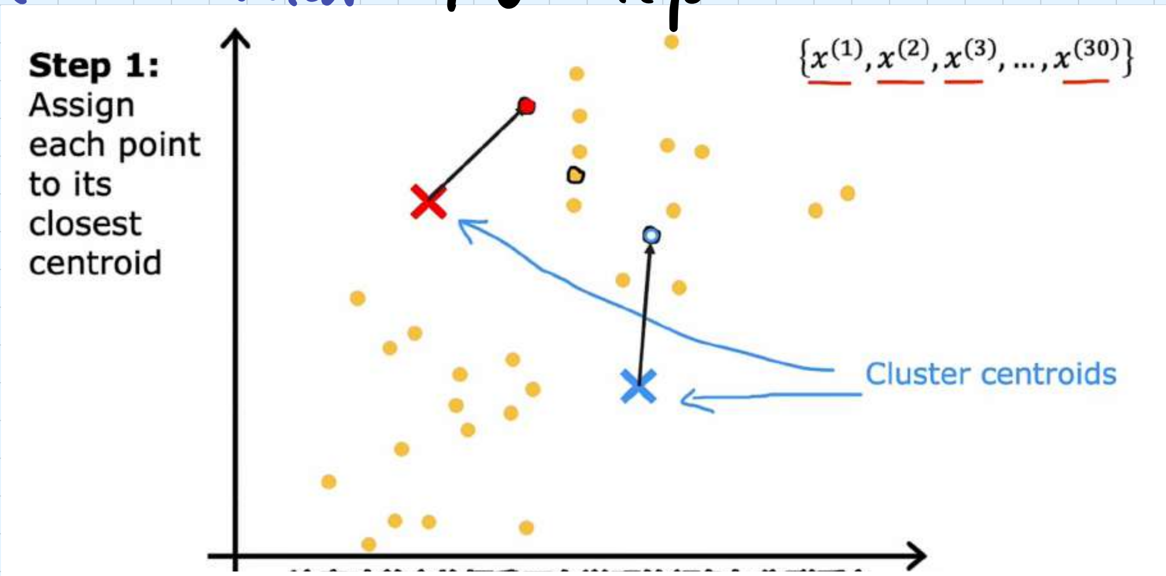
What is clustering

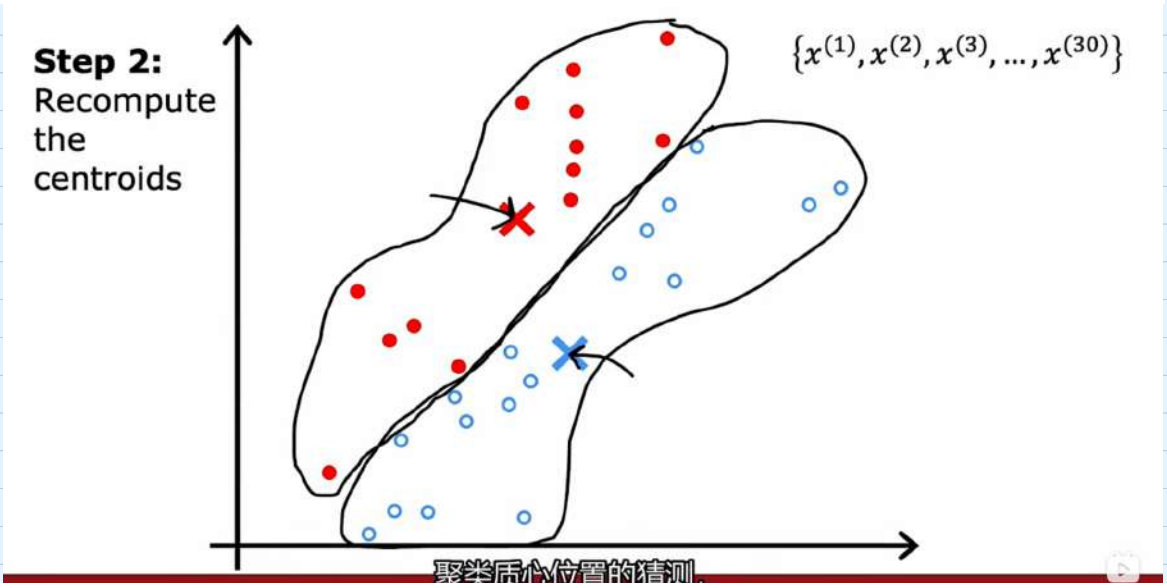


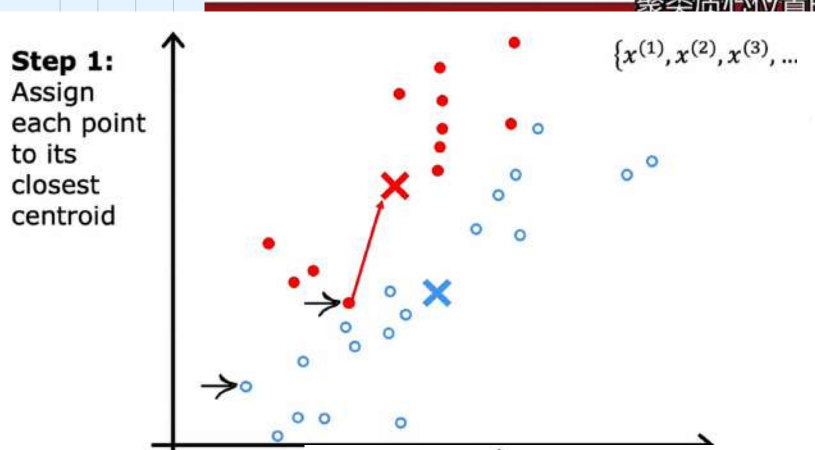


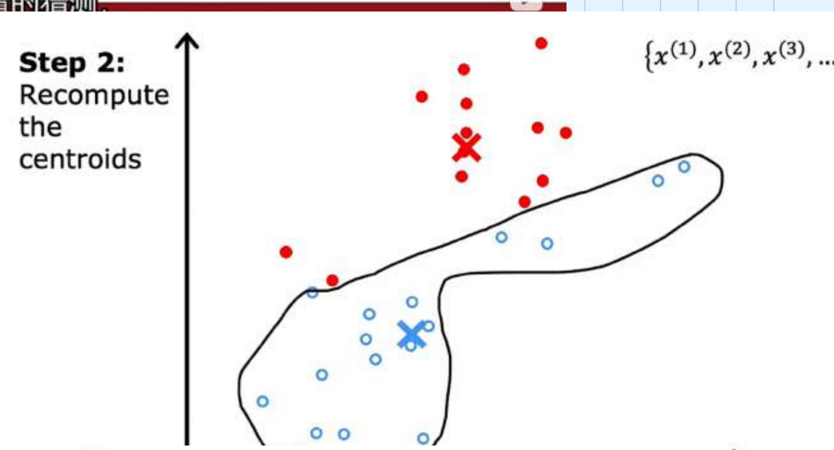


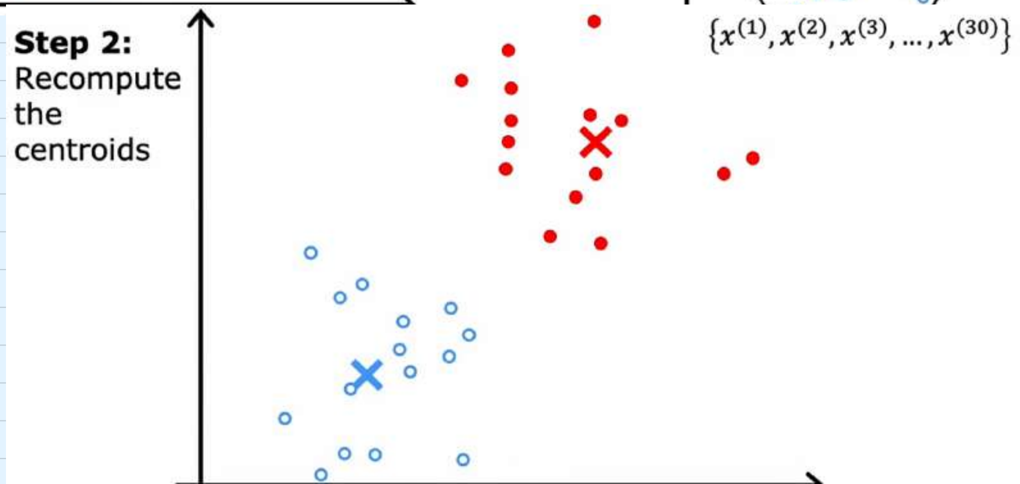
K-means intuition   Two steps











K-means algorithm

K-means algorithm

μ_1, μ_2 $x^{(1)}, x^{(2)}, \ldots x^{(30)}$

$n=2$ $K=2$

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K$

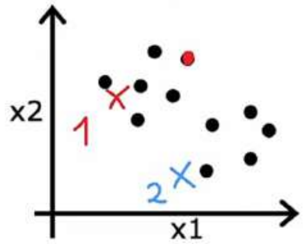
Repeat {

 # Assign points to cluster centroids

for $i = 1$ to m

$c^{(i)}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$

$\min_k \| x^{(i)} - \mu_k \|^2$



K-means algorithm

μ_1, μ_2 $x^{(1)}, x^{(2)}, \ldots x^{(30)}$

$\boxed{n=2}$ $K=2$

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K$

Repeat {

 # Assign points to cluster centroids

for $i = 1$ to m

$c^{(i)}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$

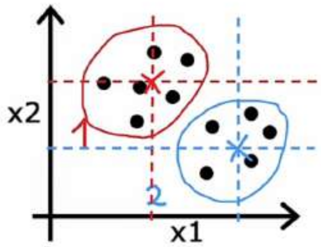
 # Move cluster centroids

for $k = 1$ to K

μ_k := average (mean) of points assigned to cluster k

}

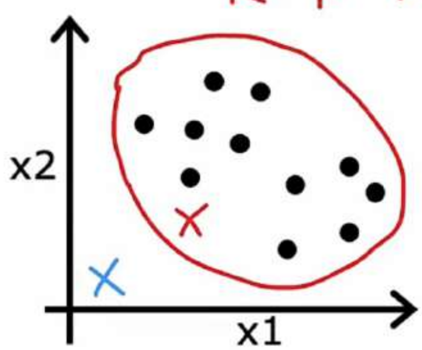
$$\mu_1 = \frac{1}{4}\left[x^{(1)} + x^{(5)} + x^{(6)} + x^{(10)}\right]$$



$n=2$ $\cancel{K=2}$

$, \ldots, \mu_K$

$K = K-1$ ✓

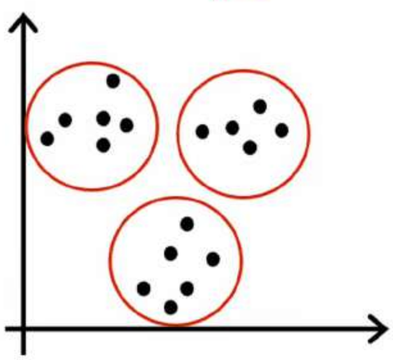


ter k

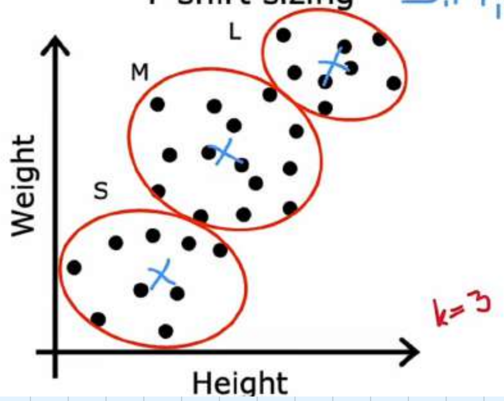
遇到这问题
直接删K

K-means for clusters that are not well separated

$k=3$



T-shirt sizing S, M, L



↳ 在分类还是不行

Optimization objective 优化目标

K-means optimization objective

$c^{(i)}$ = index of cluster $(1, 2, \ldots, K)$ to which example $x^{(i)}$ is currently assigned

μ_k = cluster centroid k

$\mu_{c^{(i)}}$ = cluster centroid of cluster to which example $x^{(i)}$ has been assigned

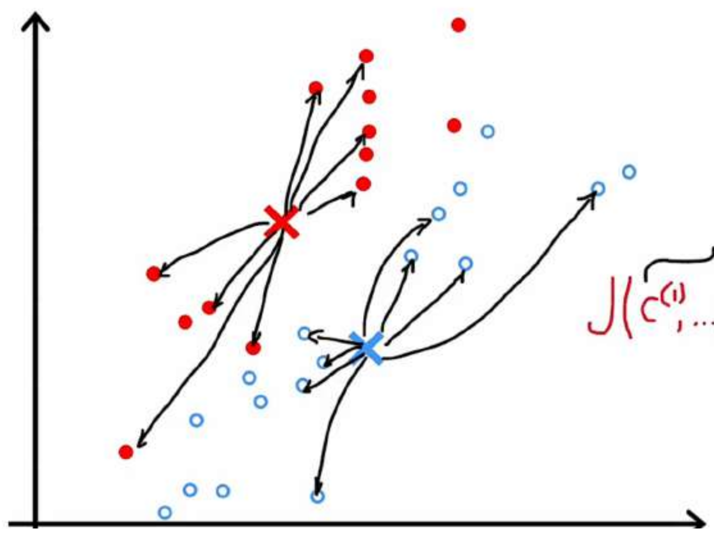
$x^{(10)}$ $c^{(10)}$ $\mu_c^{(10)}$

Cost function

$$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K) = \frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)} - \mu_{c^{(i)}}\right\|^2$$

$$\min_{\substack{c^{(1)}, \ldots, c^{(m)} \\ \mu_1, \ldots, \mu_K}} J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$$

Distorian



$J(\overbrace{c^{(1)}, \ldots, c^{(30)}}, \underbrace{\mu_1, \mu_2})$

Cost function for K-means

$$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K) = \frac{1}{m}\sum_{i=1}^{m}\boxed{\left\|x^{(i)} - \mu_{c^{(i)}}\right\|^2}$$

Repeat {

 # Assign points to cluster centroids

for i = 1 to m

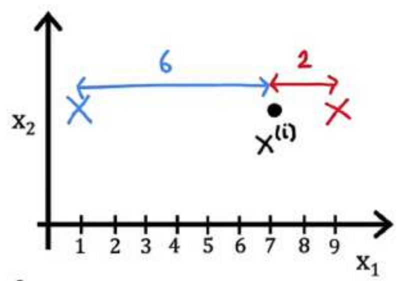
$c^{(i)}$:= index of cluster centroid closest to $x^{(i)}$

 # Move cluster centroids

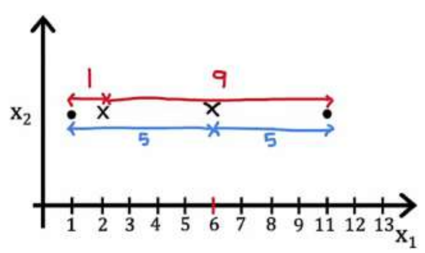
for k = 1 to K

μ_k := average of points in cluster k

}



Moving the centroid



$\frac{1}{2}(1^2 + 9^2) = \frac{1}{2}(1+81) = 41$

$\frac{1}{2}(1 + 11) = 6$

$\frac{1}{2}(5^2 + 5^2) = 25$

Initializing K-means

K-means algorithm

Step 0: Randomly initialize K cluster centroids $\mu_1, \mu_1, \ldots, \mu_k$

Repeat {

Step 1: Assign points to cluster centroids

Step 2: Move cluster centroids

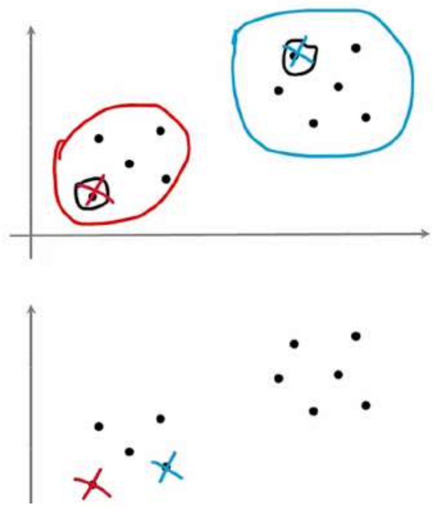
}

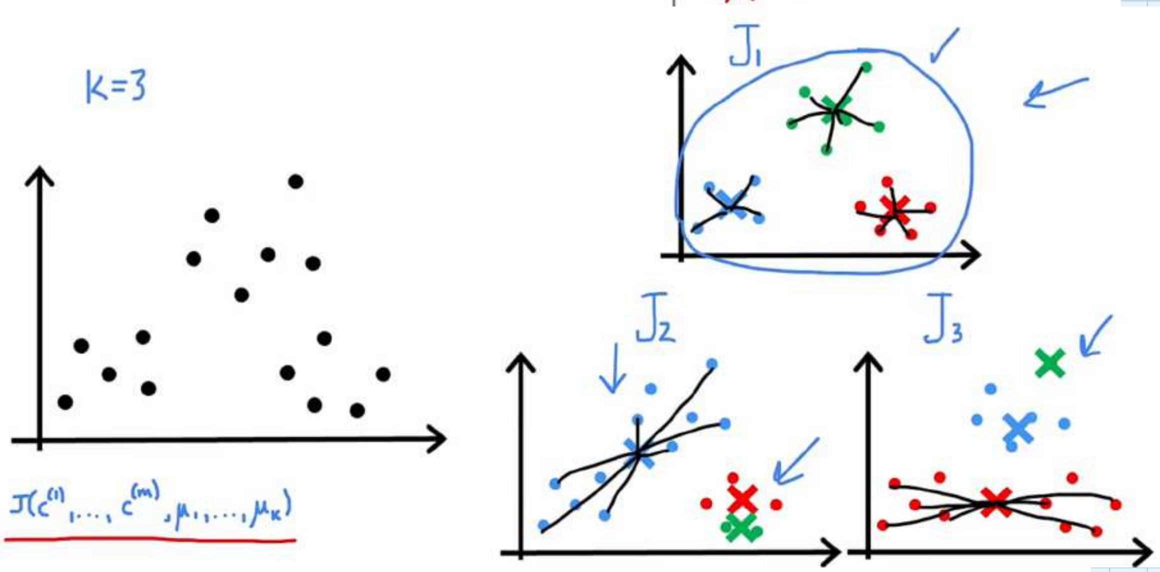
Random initialization

Choose $K < m$

Randomly pick K training examples.

Set $\mu_1, \mu_1, \ldots, \mu_k$ equal to these K examples.





多跑

找

最优

局部解

Random initialization

For $i = 1$ to 100 { ← 50-1000

Randomly initialize K-means. ← k random examples

Run K-means. Get $c^{(1)}, \ldots, c^{(m)}, \mu_1, \mu_1, \ldots, \mu_k$ ←

Computer cost function (distortion)

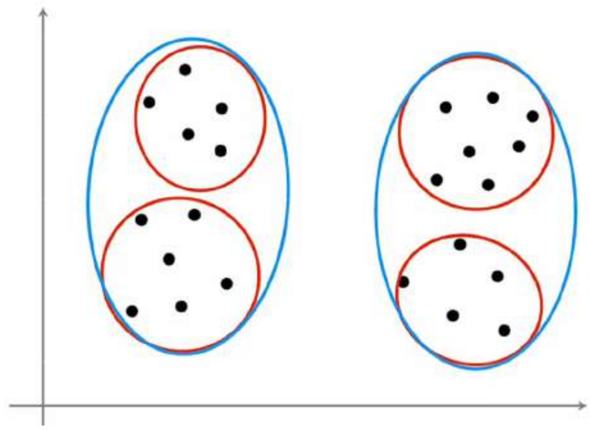
$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \mu_1, \ldots, \mu_k)$ ←

}

Pick set of clusters that gave lowest cost J

Choosing the number of clusters

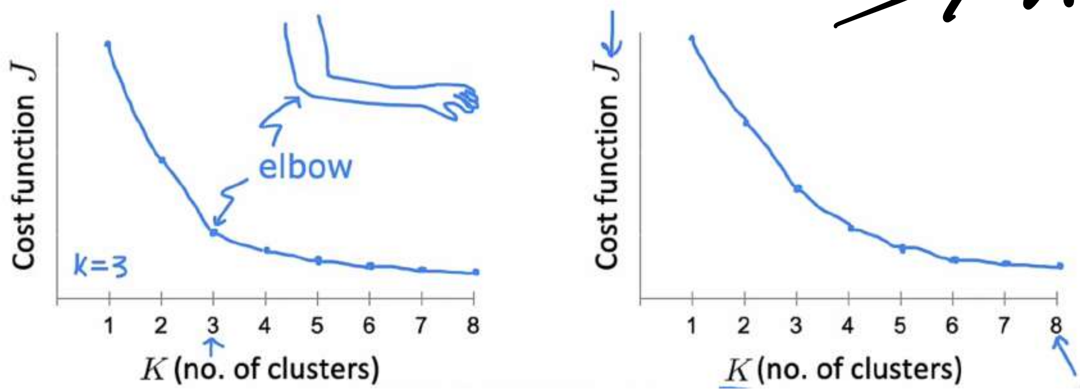
What is the right value of K?



Choosing the value of K

Elbow method:

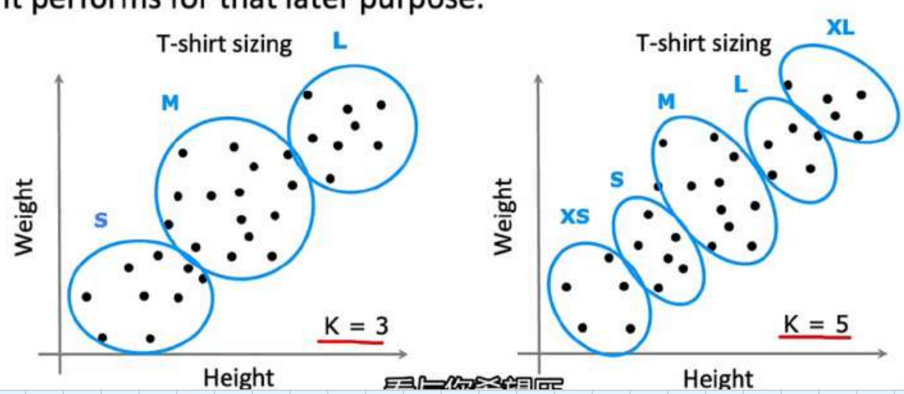
→ 不推荐



Choosing the value of K

Sometimes, you're running K-means to get clusters to use for some later/downstream purpose. Evaluate K-means based on a metric for how well it performs for that later purpose.

E.g.



↳ 根据需求选择K，
两个都跑才评估

Second algorithms ~ Anomaly Detection

Finding unusual events 异常检测

Anomaly detection example

Aircraft engine features:

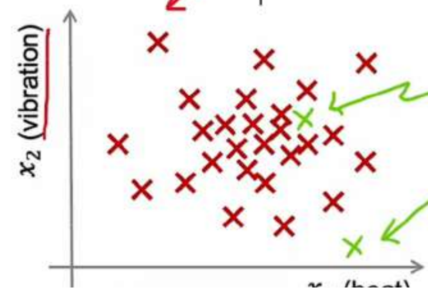
x_1 = heat generated

x_2 = vibration intensity

...

Dataset: $\{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$

New engine: x_{test}

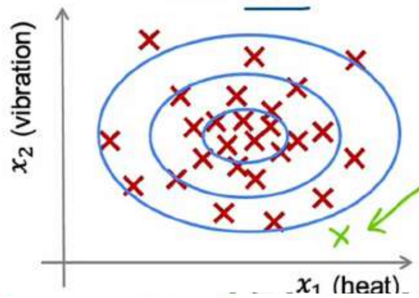


Density estimation

Dataset: $\{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$ ←

Is x_{test} anomalous?

Model p(x)



$p(x_{test}) < \varepsilon \rightarrow$ Flag anomaly

$p(x_{test}) \geq \varepsilon \rightarrow$ "ok"

Anomaly detection example

Fraud detection:

→ $x^{(i)}$ = features of user i's activities

Model $p(x)$ from data.

Identify unusual users by checking which have $p(x) < \varepsilon$

Monitoring computers in a data center.

$x^{(i)}$ = features of machine i

x_1 = memory use, x_2 = number of disk accesses/sec,

x_3 = CPU load, x_4 = CPU load/network traffic.

...

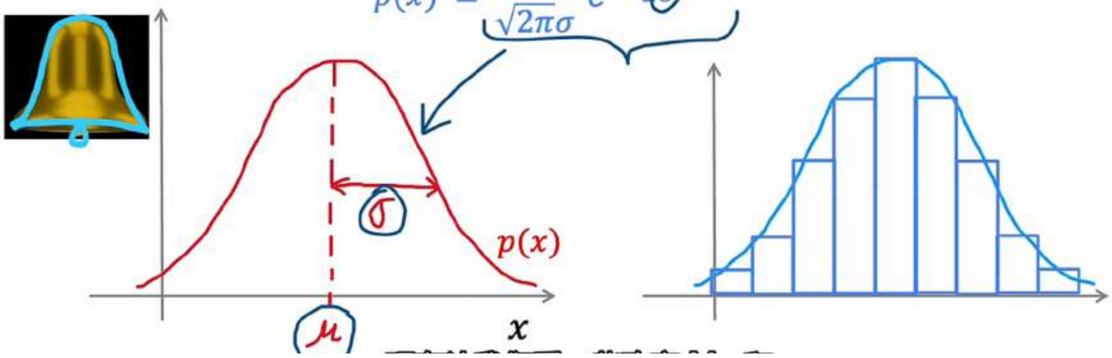
↳ 可以做防欺诈

Gaussian (Normal) Distribution

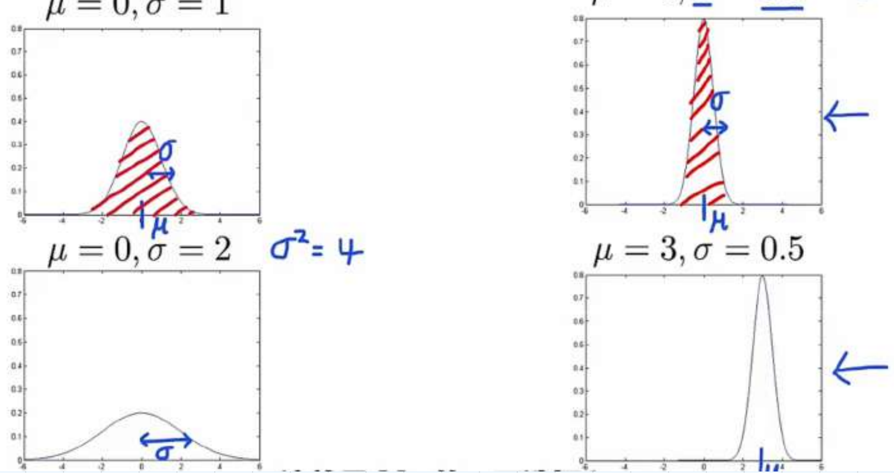
Gaussian (Normal) distribution

Say x is a number. If x is a distributed Gaussian with mean μ, σ^2 variance .

$$p(x) = \underbrace{\frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(x-\mu)^2}{2\sigma^2}}}$$

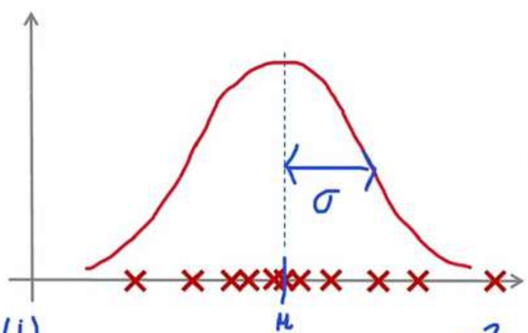


Gaussian distribution example



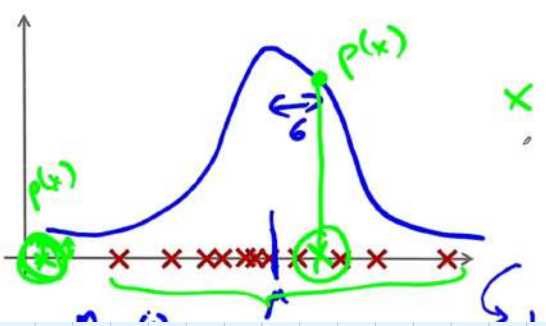
Parameter estimation

Dataset: $\{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$



$\frac{1}{m-1}$

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)} \qquad \sigma^2 = \frac{1}{m}\sum (x^{(i)} - \mu)^2$$

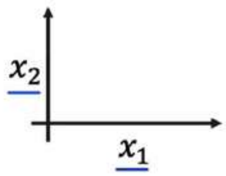


Algorithm

Density estimation

Training set: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

Each example x_i has n features



$$x = \begin{bmatrix} x_1 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$$

$$p(x) = p(x_1; \mu_1, \sigma_1^2) * p(x_2; \mu_2, \sigma_2^2) * p(x_3; \mu_3, \sigma_3^2) * p(x_n; \mu_n, \sigma_n^2)$$

$$= \prod_{j=1}^{n} p(x_j; \mu_j, \sigma_j^2)$$

$\sum \to +$ $\prod \to \times$

$p(x_1 = high\ temp) = 1/10$

$p(x_2 = high\ vibra) = 1/20$

$p(x_1, x_2) = p(x_1) * p(x_2)$

$= \frac{1}{10} * \frac{1}{20} = \frac{1}{200}$

Anomaly detection algorithm

1. Choose n features x_i that you think might be indicative of anomalous examples.
2. Fit parameters $\mu_1, \ldots, \mu_n, \sigma_1^2, \ldots, \sigma_n^2$

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)} \qquad \sigma_j^2 = \frac{1}{m}\sum_{i=1}^{m}(x_j^{(i)} - \mu_j)^2$$

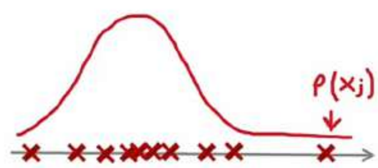
Vectorized formula

$$\vec{\mu} = \frac{1}{m}\sum_{i=1}^{m} \vec{x}^{(i)} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \cdots \\ \mu_n \end{bmatrix}$$

Anomaly detection algorithm

1. Choose n features x_i that you think might be indicative of anomalous examples.
2. Fit parameters $\mu_1, \ldots, \mu_n, \sigma_1^2, \ldots, \sigma_n^2$

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)} \qquad \sigma_j^2 = \frac{1}{m}\sum_{i=1}^{m}(x_j^{(i)} - \mu_j)^2$$

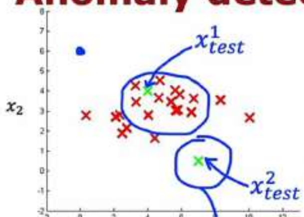


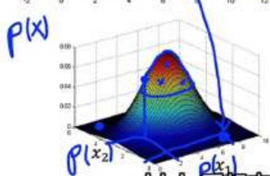
3. Given new example x, compute $p(x)$:

$$p(x) = \prod_{j=1}^{n} \boxed{p(x_j; \mu_j, \sigma_j^2)} = \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_j} exp\left(-\frac{(x_j - \mu_j)^2}{2\sigma_j^2}\right)$$

Anomaly if $p(x) < \varepsilon$

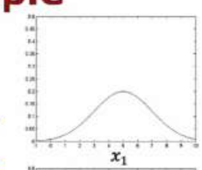
Anomaly detection example



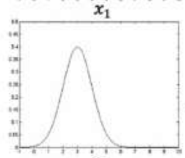


$\mu_1 = 5, \sigma_1 = 2$

$\mu_2 = 3, \sigma_2 = 1$



$p(x_1; \mu_1, \sigma_1^2)$



$p(x_2; \mu_2, \sigma_2^2)$

$\varepsilon = 0.02$

$p(x_{test}^{(1)}) = 0.0426 \xrightarrow{x_2}$ "ok"

$p(x_{test}^{(2)}) = 0.0021 \longrightarrow$ "anomaly"

Developing and evaluating an anomoly detection system

The importance of real-number evaluation

When developing a learning algorithm (choosing features, etc.), making decisions is much easier if we have a way of evaluating our learning algorithm.

Assume we have some labeled data, of anomalous and non-anomalous examples. ($y = 0$ if normal, $y = 1$ if anomalous).

Training set: $x^{(1)}, x^{(2)}, \dots, x^{(m)}$ (assume normal examples/not anomalous) $y=0$

Cross validation set: $\left(x_{cv}^{(1)}, y_{cv}^{(1)}\right), \dots, \left(x_{cv}^{(m_{cv})}, y_{cv}^{(m_{cv})}\right)$

Test set: $\left(x_{test}^{(1)}, y_{test}^{(1)}\right), \dots, \left(x_{test}^{(m_{test})}, y_{test}^{(m_{test})}\right)$

Include a few anomalous examples $y=1$

Aircraft engines monitoring example

10000 good (normal) engines 2-50

20 2 flawed engines (anomalous) $y=1$

Training set: 6000 good engines Train algorithm $y=0$

CV: 2000 good engines ($y = 0$), 10 anomalous ($y = 1$) ε x_j

Test: 2000 good engines ($y = 0$), 10 anomalous ($y = 1$)

Alternative:

Training set: 6000 good engines 2

CV: 4000 good engines ($y = 0$), 20 anomalous ($y = 1$)

No test set ε, x_j

Algorithm evaluation

Fit model $p(x)$ on training set $x^{(1)}, x^{(2)}, \dots, x^{(m)}$

On a cross validation/test example x, predict

$$y = \begin{cases} 1 & if\ \boxed{p(x)} < \varepsilon\ (anomaly) \\ 0 & if\ p(x) \geq \varepsilon\ (normal) \end{cases}$$

10

2000

Possible evaluation metrics:

- True positive, false positive, false negative, true negative
- Precision/Recall
- F_1-score

- F_1-score

Can also use cross validation set to choose parameter ε

Andrew

Anomaly detection VS Supervised learning

Anomaly detection vs. Supervised learning

Anomaly detection	Supervised learning
Very small number of positive examples ($y = 1$). (0-20 is common). Large number of negative ($y = 0$) examples. p(x)	Large number of positive and negative examples.
y=1	20 positive examples
Many different "types" of anomalies. Hard for any algorithm to learn from positive examples what the anomalies look like; future anomalies may look nothing like any of the anomalous examples we've seen so far.	Enough positive examples for algorithm to get a sense of what positive examples are like, future positive examples likely to be similar to ones in training set.
Fraud	Spam

新欺诈类型，则可能更适用。

↳ 防欺诈

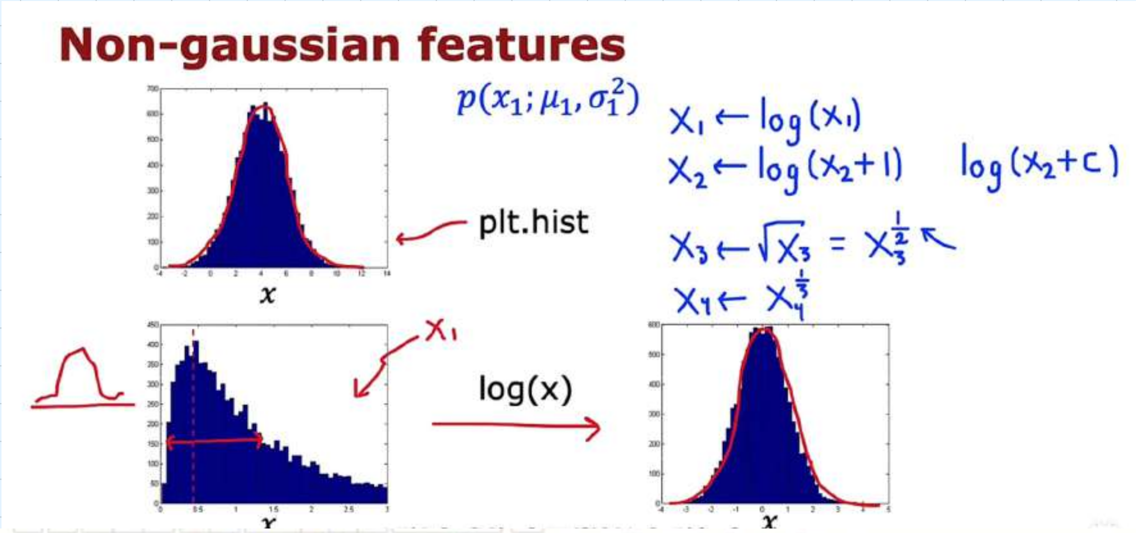
Anomaly detection vs. Supervised learning

Anomaly detection	Supervised learning
→ Fraud detection	→ Email spam classification
→ • Manufacturing - Finding new previously unseen defects in manufacturing.(e.g. aircraft engines)	→ Manufacturing - Finding known, previously seen defects y=1 scratches
→ Monitoring machines in a data center	→ Weather prediction (sunny/rainy/etc.)
⋮	→ Diseases classification
	⋮

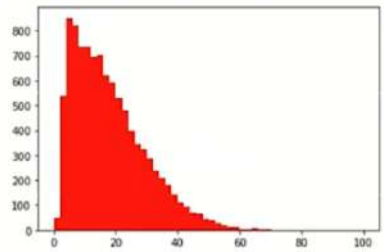
↓ 检测新缺陷

↑ similiar old and future

Choosing what features to use



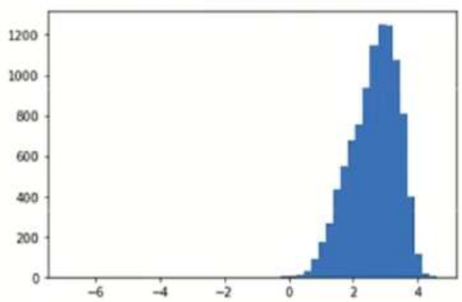

```
random = random - min(random)        #Shift the set so the minimum value is equal to zero.
random = random / max(random)        #Standadize all the vlues between 0 and 1.
random = random * maxValue           #Multiply the standardized values by the maximum value.

x = random

#Plot histogram to check skewness
plt.hist(x, bins=50, color='r');
```



```
In [5]:  plt.hist(x**0.5, bins=50);
```

再次发出嘶嘶声的一克垃圾箱，在这种情况下，它可能看起来像这样。
in which case it might look like this.

```
In [11]:  plt.hist(np.log(x+0.001), bins=50);
```



```
In [10]:  np.min(x)
Out[10
```

所以你会得到看起来像这样的嘶嘶声。

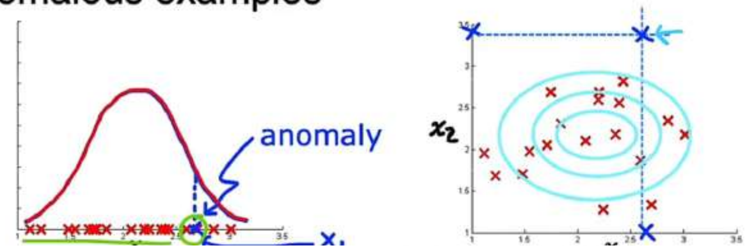
```
In [16]:  plt.hist(np.log(x+7), bins=50);
```

因此，通过在 Jupiter notebook

Error analysis for anomaly detection

Want $p(x) \geq \varepsilon$ large for normal examples x.
$p(x) < \varepsilon$ small for anomalous examples x.

Most common problem:
$p(x)$ is comparable (say, both large) for normal and anomalous examples



Monitoring computers in a data center

Choose features that might take on unusually large or small values in the event of an anomaly.

x_1 = memory use of computer
x_2 = number of disk accesses/sec
x_3 = CPU load
x_4 = network traffic

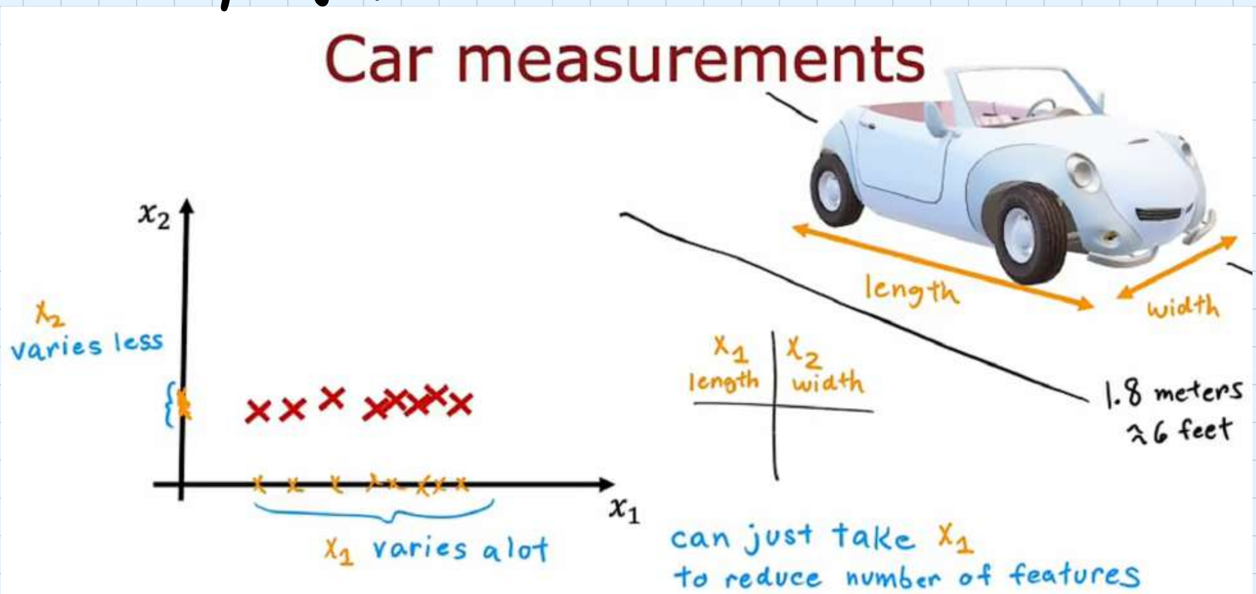
$$x_5 = \frac{\text{CPU load}}{\text{network traffic}} \qquad x_6 = \frac{(\text{CPU load})^2}{\text{network traffic}}$$

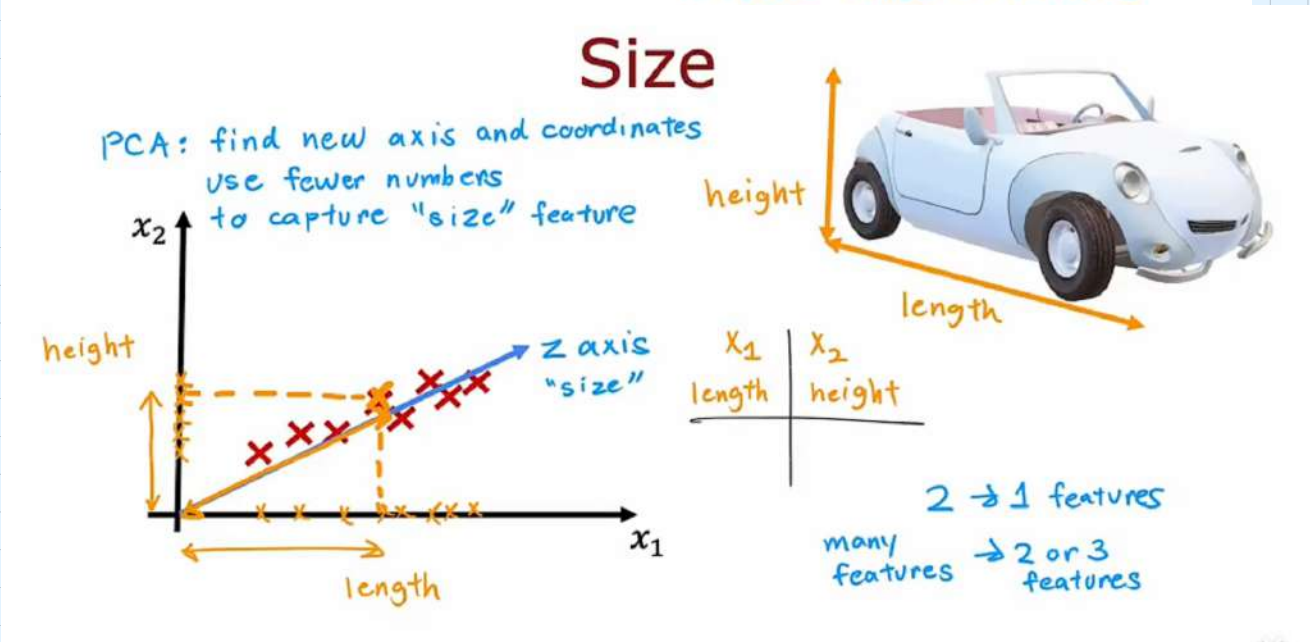
Deciding feature choice based on $p(x)$
– large for normal examples, and becomes small for anomaly in the cross validation set

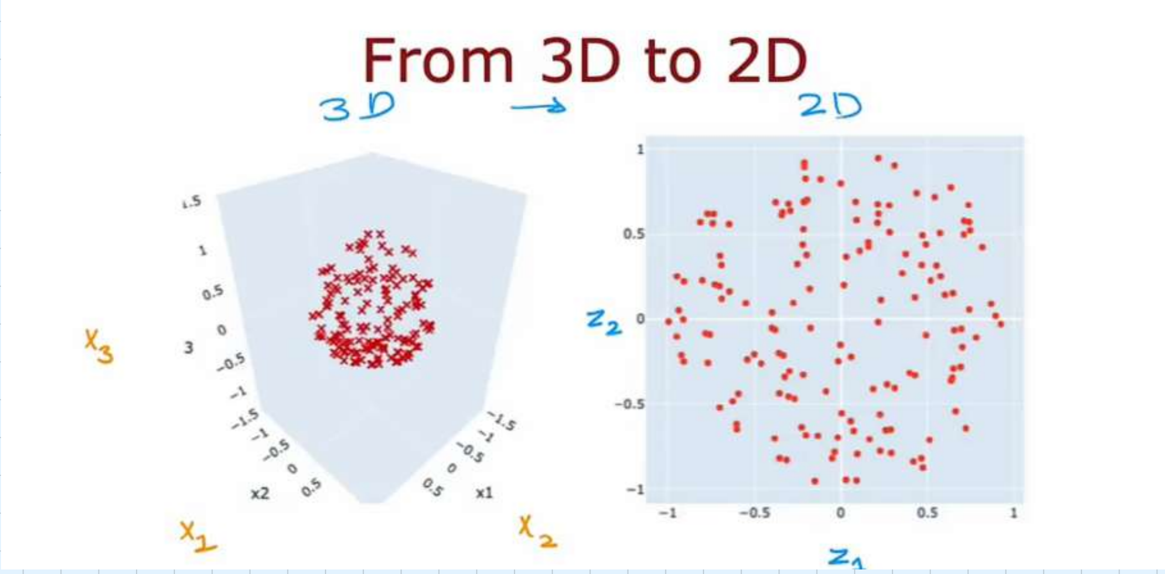
选修

主成分分析 (PCA算法)

↳ 常用来可视化





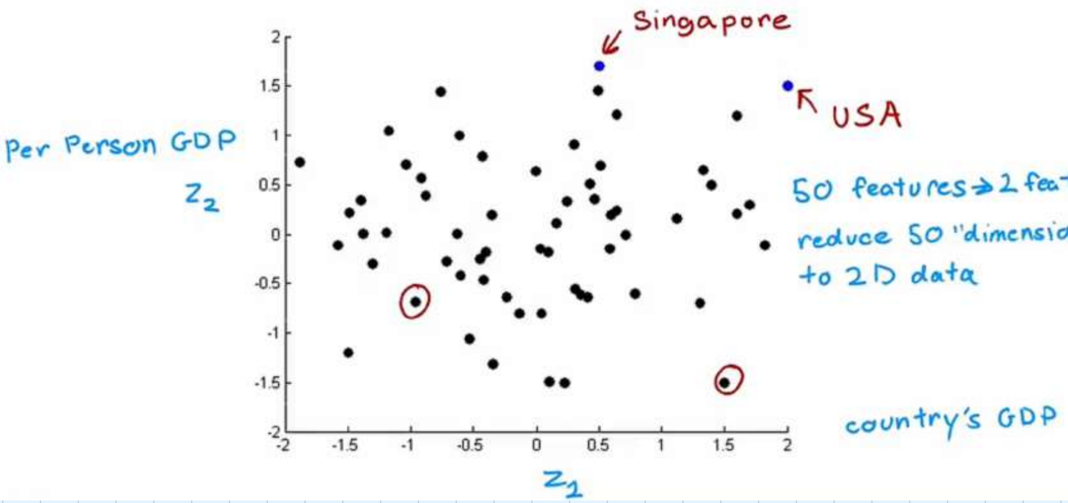


50 features

Country	x_1 GDP (trillions of US$)	x_2 Per capita GDP (thousands of intl. $)	x_3 Human Development Index	x_4 Life expectancy	••• Poverty Index (Gini as percentage)	Mean household income (thousands of US$)	…
Canada	1.577	39.17	0.908	80.7	32.6	67.293	…
China	5.878	7.54	0.687	73	46.9	10.22	…
India	1.632	3.41	0.547	64.7	36.8	0.735	…
Russia	1.48	19.84	0.755	65.5	39.9	0.72	…

Country	z_1	z_2
Canada	1.6	1.2
China	1.7	0.3
India	1.6	0.2
Russia	1.4	0.5

Data visualization



50 features → 2 features

reduce 50 "dimensional" data to 2D data

PCA algorithm

PCA algorithm

coordinates

$x_1 = 10$ $x_2 = 8$

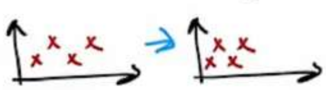
Can we choose a different axis?

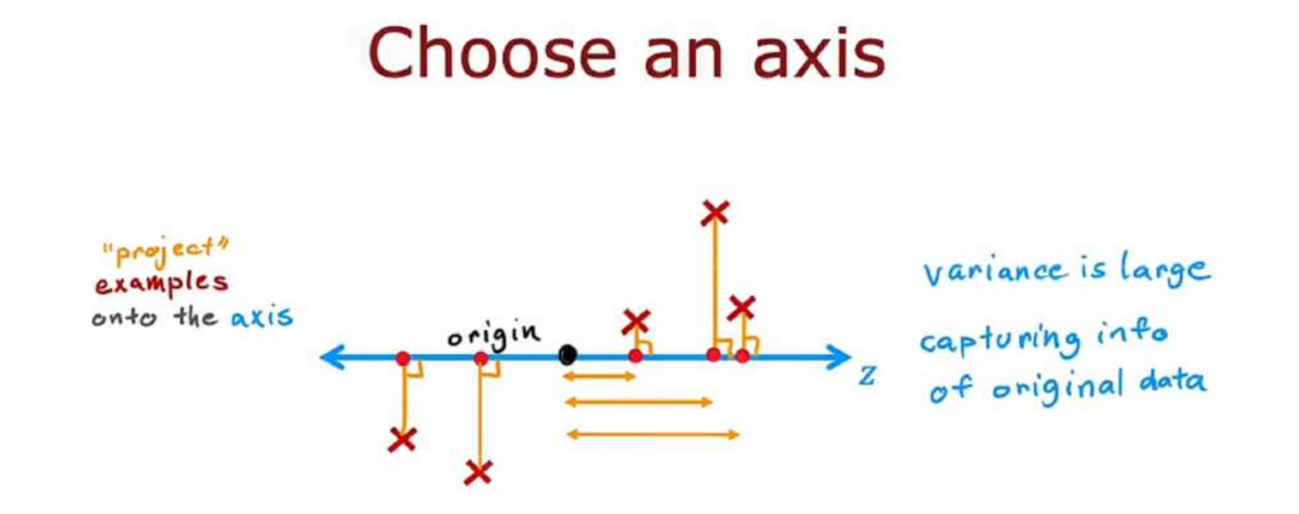
housing example: course 1 week 2

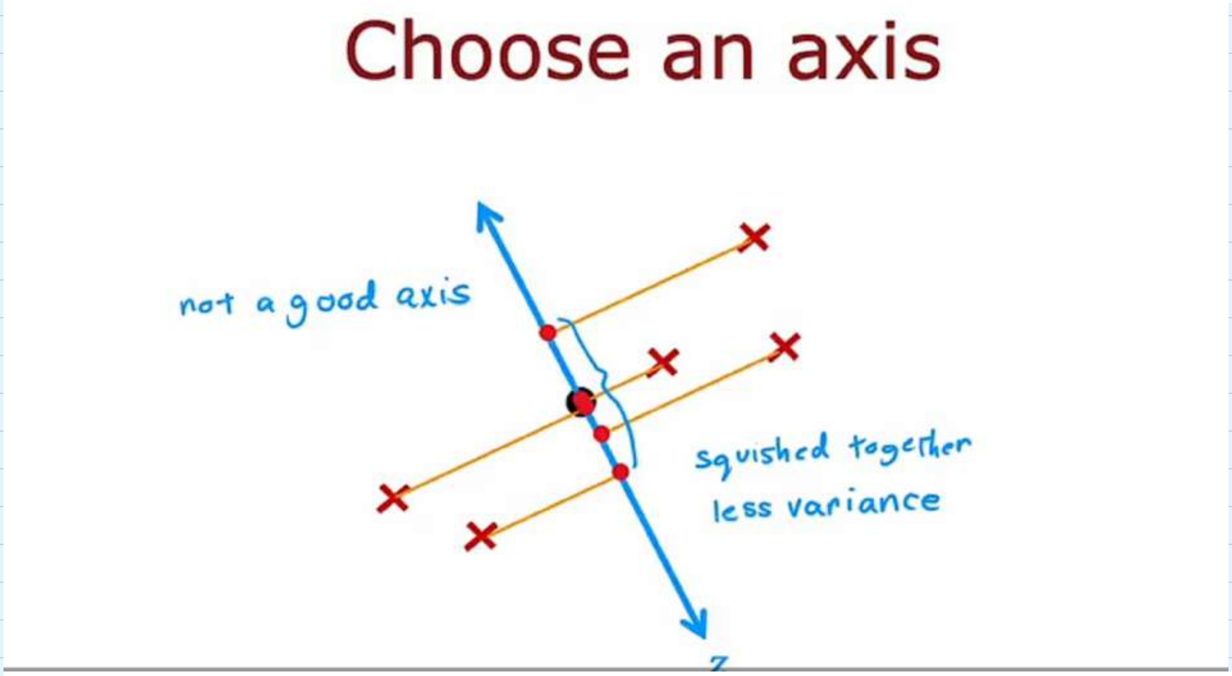
Preprocess features

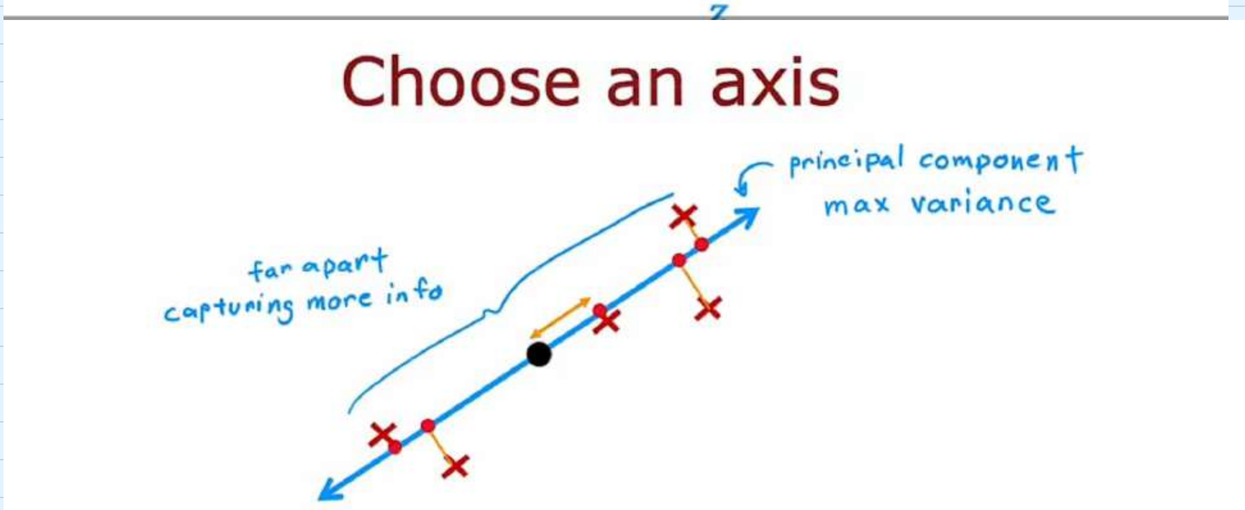
Normalized to have zero mean

feature scaling



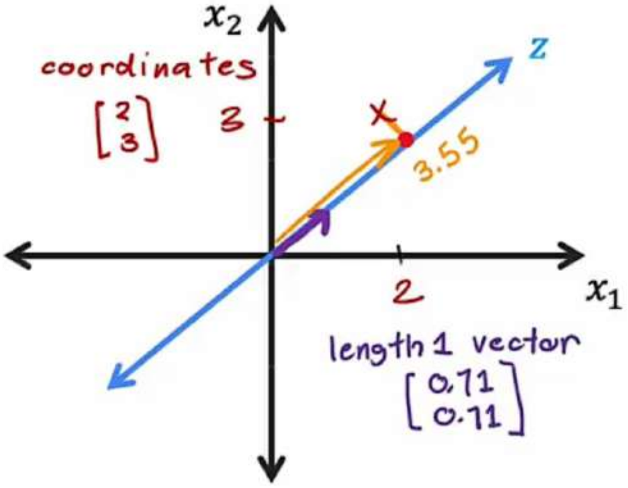

Choose an axis
"project"
examples
onto the axis
origin
z
variance is large
capturing info
of original data


Choose an axis
not a good axis
squished together
less variance
z


Choose an axis
principal component
max variance
far apart
capturing more info

↳ 最好的选择

Coordinate on the new axis

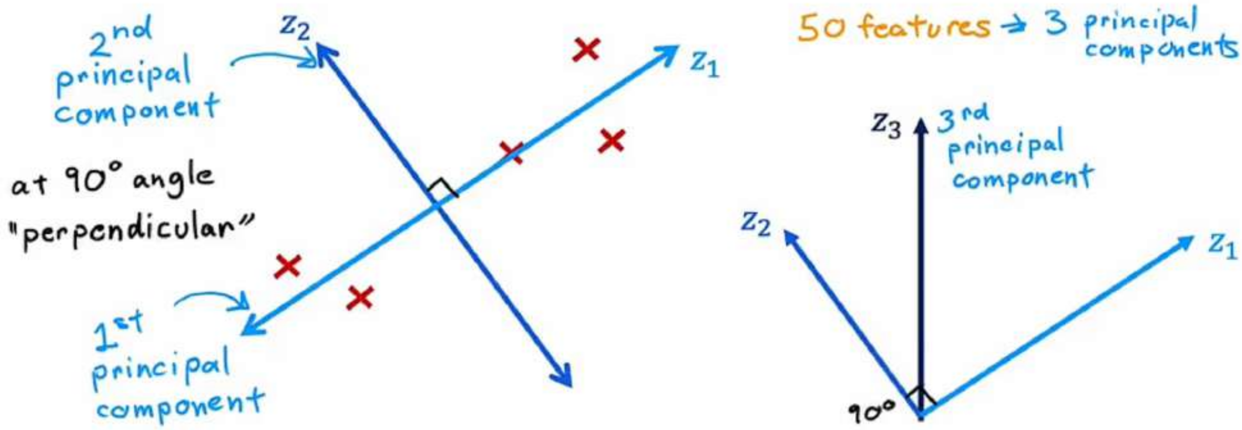


dot product

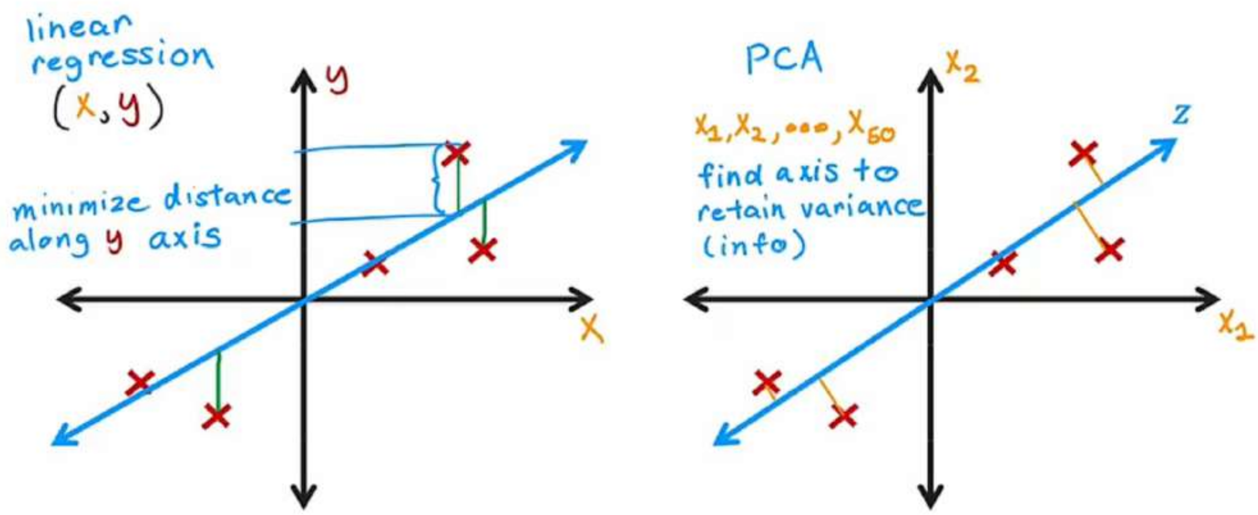
$$\begin{bmatrix}2\\3\end{bmatrix}\cdot\begin{bmatrix}0.71\\0.71\end{bmatrix}$$

$2 \times 0.71 + 3 \times 0.71 = 3.55$

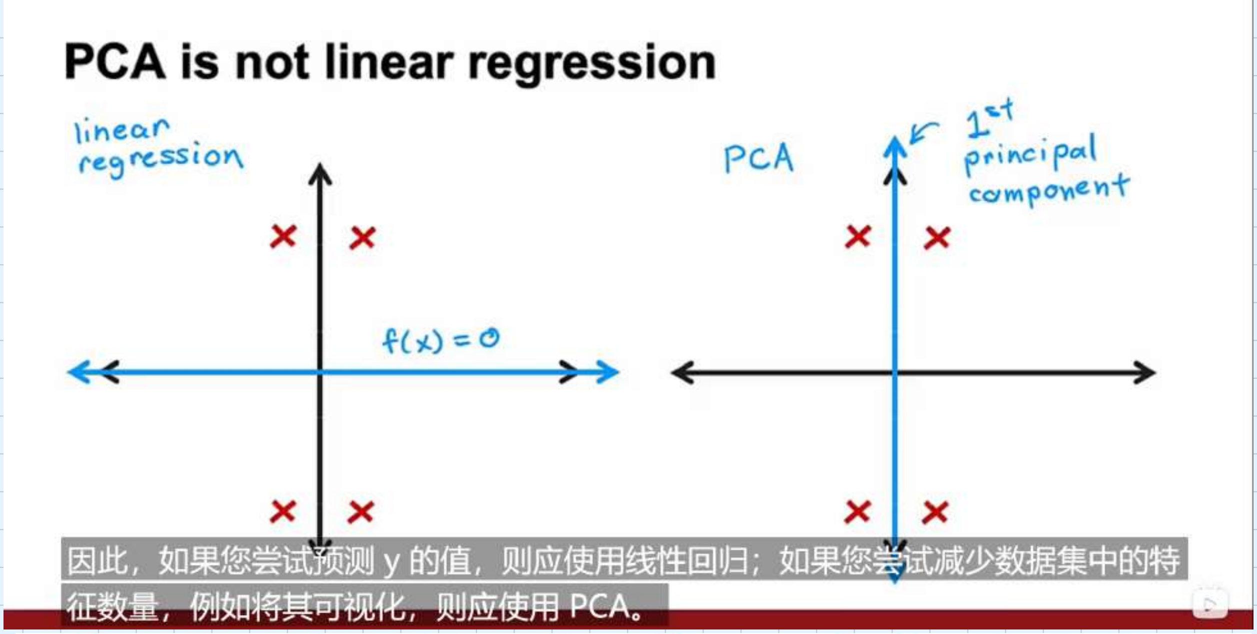
More principal components

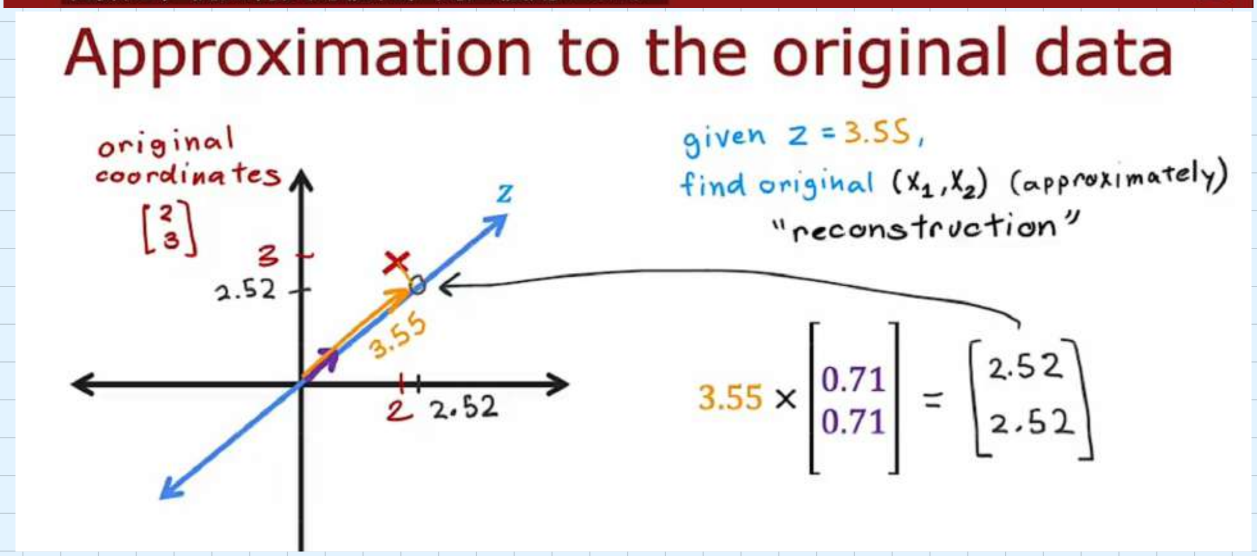


PCA is not linear regression

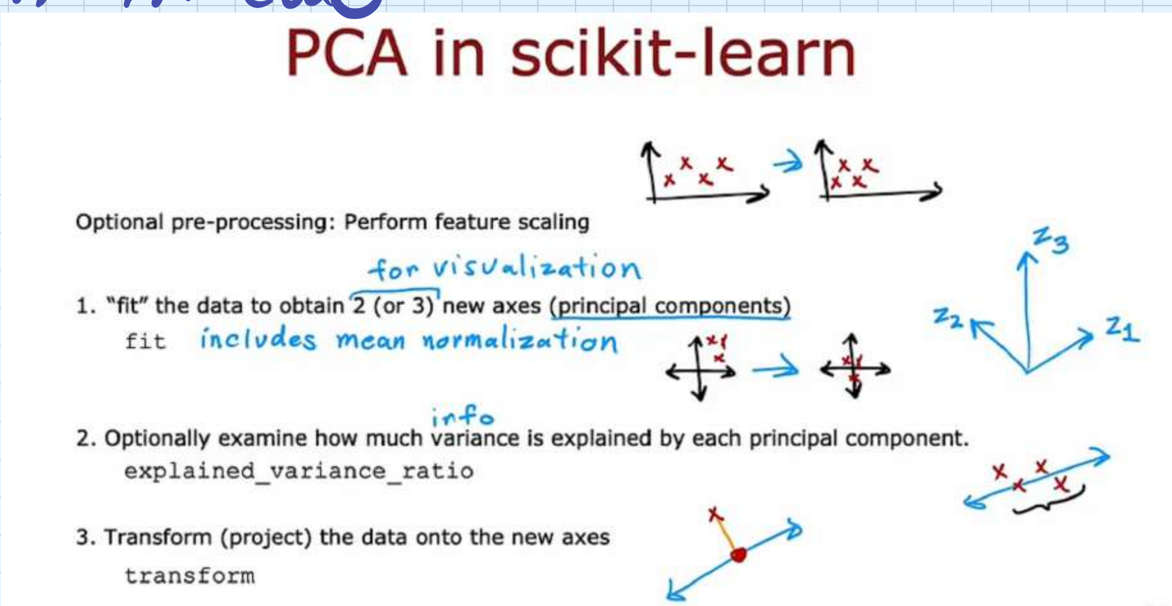


不一样

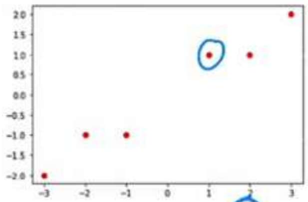

PCA is not linear regression
linear regression
f(x) = 0
PCA
1st principal component
因此，如果您尝试预测 y 的值，则应使用线性回归；如果您尝试减少数据集中的特征数量，例如将其可视化，则应使用 PCA。


Approximation to the original data
original coordinates
[2 3]
3
2.52
z
3.55
2
2.52
given z = 3.55,
find original (x1, x2) (approximately)
"reconstruction"
3.55 × [0.71 0.71] = [2.52 2.52]

PCA In code


PCA in scikit-learn
Optional pre-processing: Perform feature scaling
for visualization
1. "fit" the data to obtain 2 (or 3) new axes (principal components)
fit includes mean normalization
z3
z2
z1
info
2. Optionally examine how much variance is explained by each principal component.
explained_variance_ratio
3. Transform (project) the data onto the new axes
transform

Example

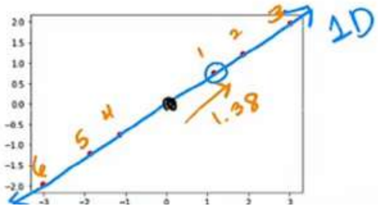


```
X = np.array([[1, 1], [2, 1], [3, 2],
              [-1, -1], [-2, -1], [-3, -2]])
```

2D

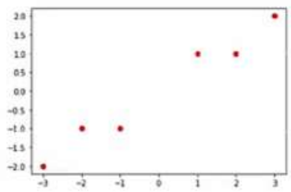
```
pca_1 = PCA(n_components=1)
pca_1.fit(X)
pca_1.explained_variance_ratio_
X_trans_1 = pca_1.transform(X)
X_reduced_1 = pca.inverse_transform(X_trans_1)
```

0.992




```
array([
1 [ 1.38340578],
2 [ 2.22189802],
3 [ 3.6053038 ],
4 [-1.38340578],
5 [-2.22189802],
6 [-3.6053038 ]])
```

Example

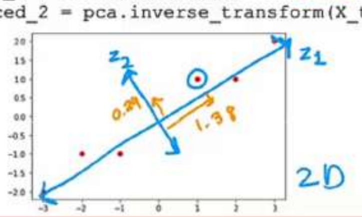


```
X = np.array([[1, 1], [2, 1], [3, 2],
              [-1, -1], [-2, -1], [-3, -2]])
```

2D

```
pca_2 = PCA(n_components=2)
pca_2.fit(X)
pca_2.explained_variance_ratio_
X_trans_2 = pca.transform(X)
X_reduced_2 = pca.inverse_transform(X_trans_2)
```

z_1 0.992 z_2 0.008




```
        z1             z2
array([
[ 1.38340578,  0.2935787 ],
[ 2.22189802, -0.25133484],
[ 3.6053038 ,  0.04224385],
[-1.38340578, -0.2935787 ],
[-2.22189802,  0.25133484],
[-3.6053038 , -0.04224385]])
```

Applications of PCA

Visualization *reduce to 2 or 3 features*

Less frequently used for:

- Data compression
  (to reduce storage or transmission costs) *50 → 10*
- Speeding up training of a supervised learning model

$n = 1000 \rightarrow 100$