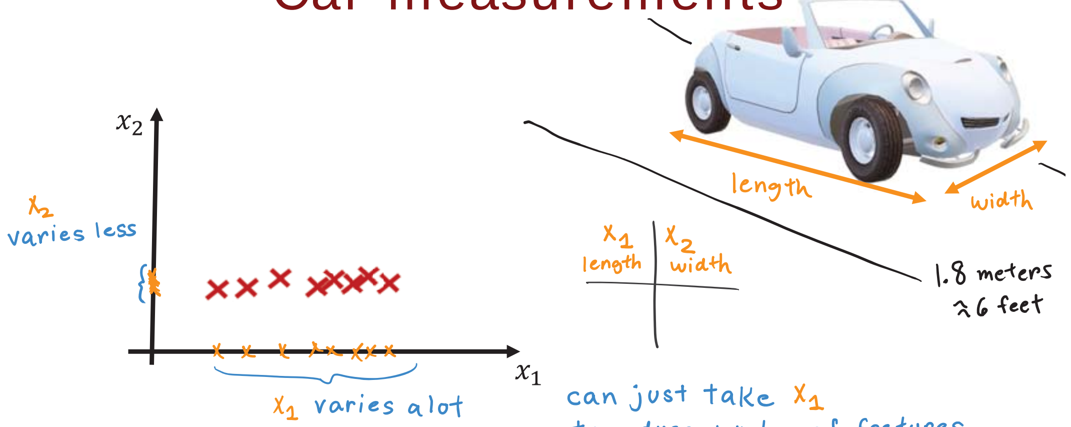
Principal Component Analysis (Optional)

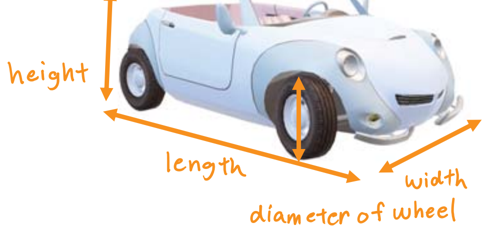
Reducing the number of features

Car measurements

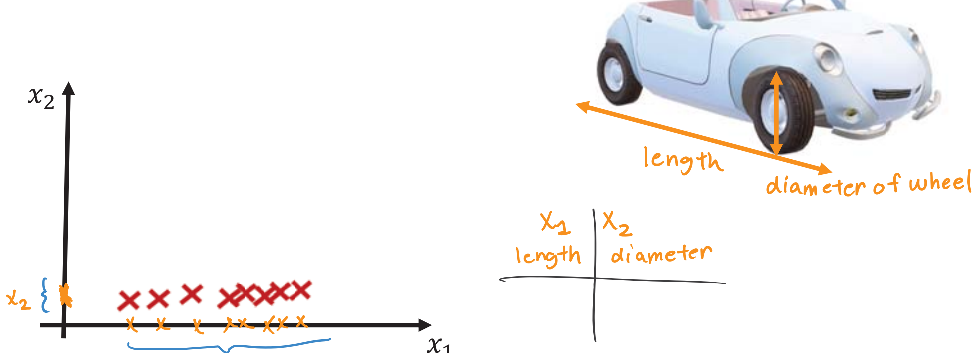


can just take x_1 to reduce number of features

Car measurements



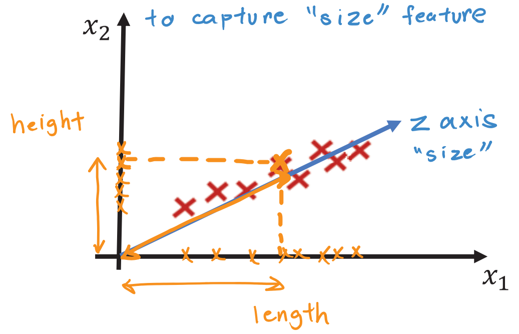
Car measurements

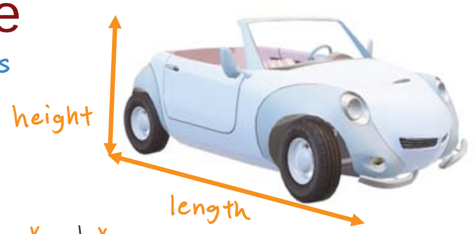


can just take x_1 to reduce number of features

Size

PCA: find new axis and coordinates
use fewer numbers
to capture "size" feature



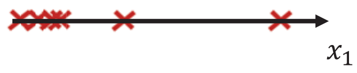


x_1	x_2
length	height

2 → 1 features

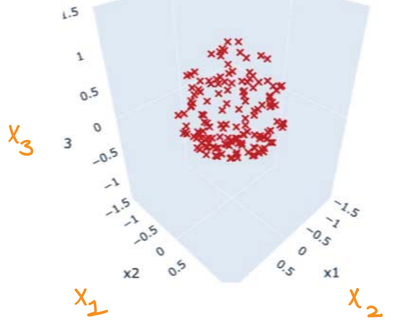
many features → 2 or 3 features

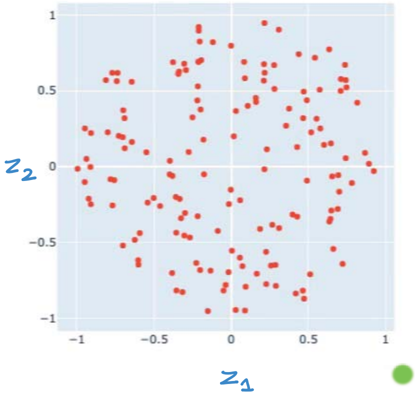
Country	GDP (trillions of US$)
Canada	1.577
China	5.878
India	1.632
Russia	1.48
Singapore	0.223
USA	14.527
…	…



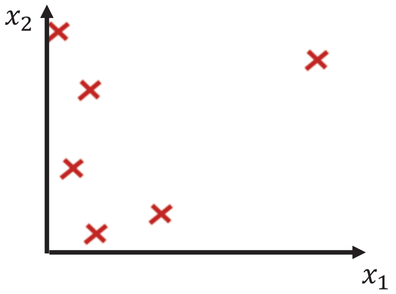
From 3D to 2D

3D → 2D

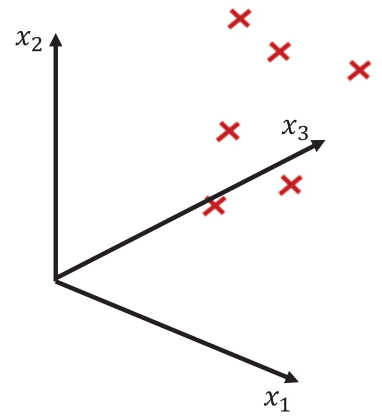




Country	GDP (trillions of US$)	Per capita GDP (thousands of intl. $)
Canada	1.577	39.17
China	5.878	7.54
India	1.632	3.41
Russia	1.48	19.84
Singapore	0.223	56.69
USA	14.527	46.86



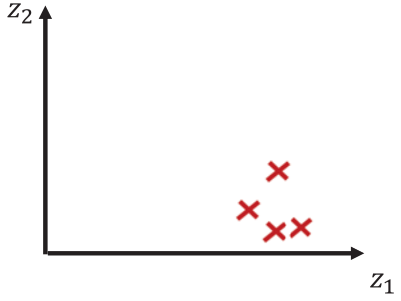
Country	GDP (trillions of US$)	Per capita GDP (thousands of intl. $)	Human Development Index
Canada	1.577	39.17	0.908
China	5.878	7.54	0.687
India	1.632	3.41	0.547
Russia	1.48	19.84	0.755
Singapore	0.223	56.69	0.866
USA	14.527	46.86	0.91
...	...	...	...



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	...

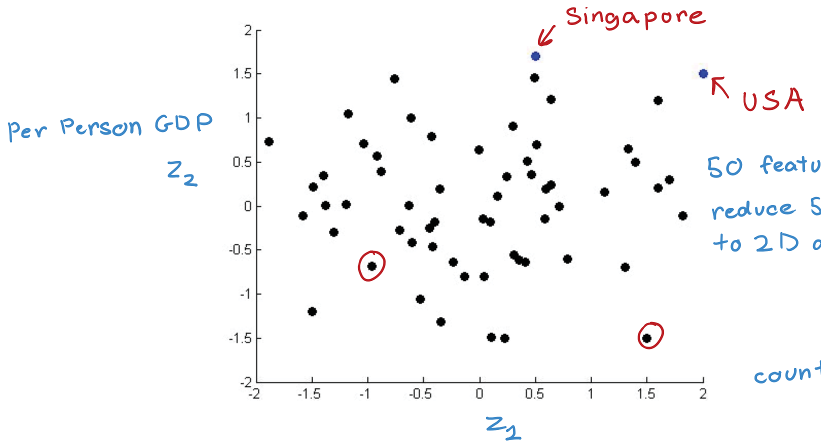
Country	z_1	z_2
Canada	1.6	1.2
China	1.7	0.3
India	1.6	0.2
Russia	1.4	0.5



Country	GDP (trillions of US$)	Per capita GDP (thousands of intl. $)	Human Development Index	Life expectancy
Canada	1.577	39.17	0.908	80.7
China	5.878	7.54	0.687	73
India	1.632	3.41	0.547	64.7
Russia	1.48	19.84	0.755	65.5
Singapore	0.223	56.69	0.866	80
USA	14.527	46.86	0.91	78.3
...	...	...	...	...

what if 50 features?

Data visualization



50 features → 2 features
reduce 50 "dimensional" data
to 2D data

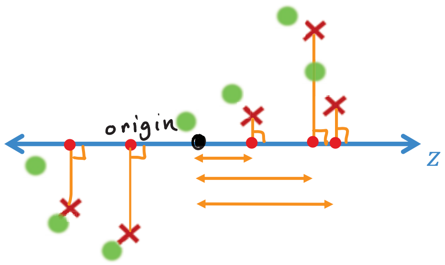
country's GDP

Principal Component Analysis

PCA Algorithm

Choose an axis

"project" examples onto the axis



variance is large

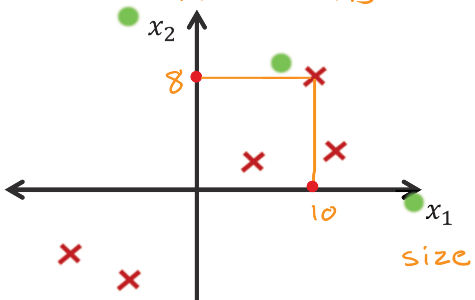
capturing info of original data

PCA algorithm

coordinates

$x_1=10$ $x_2=8$

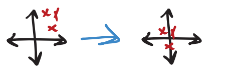
can we choose a different axis?



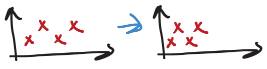
housing example: course 1 week 2

Preprocess features

Normalized to have zero mean

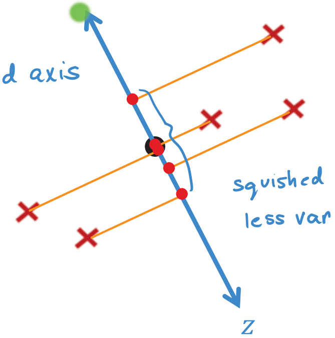


feature scaling



Choose an axis

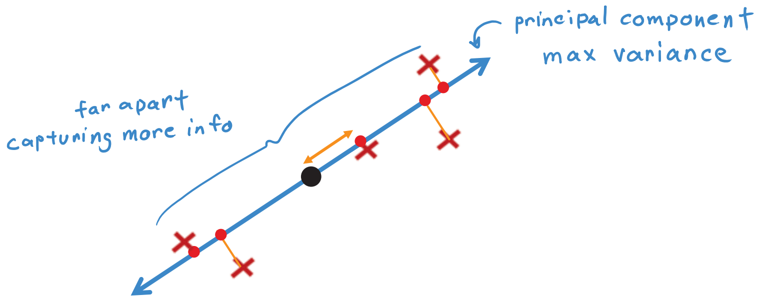
not a good axis



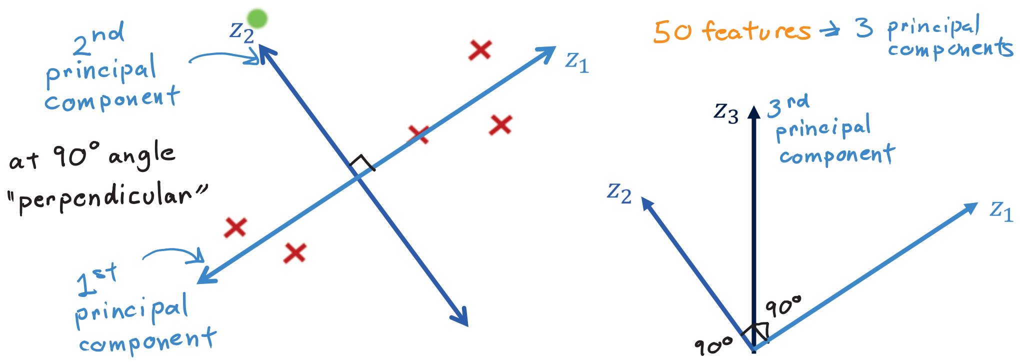
squished together

less variance

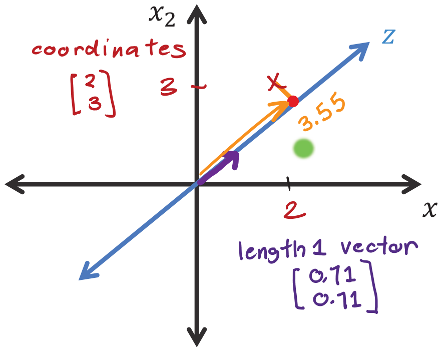
Choose an axis



More principal components



Coordinate on the new axis

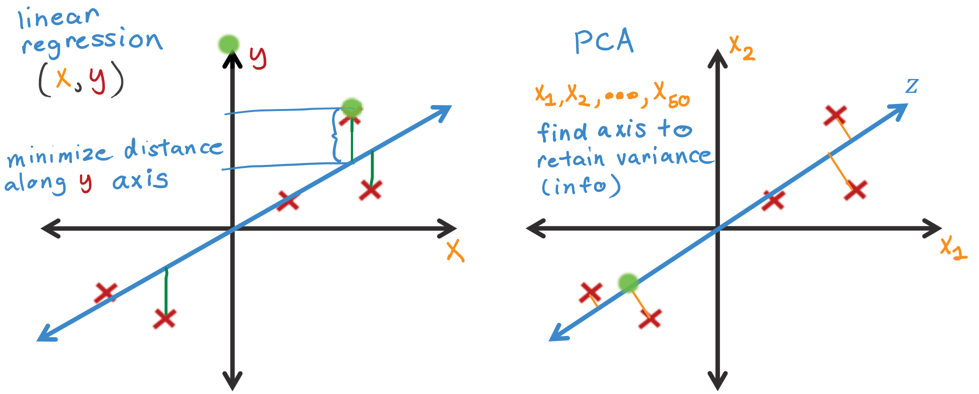

x_2
coordinates
$\begin{bmatrix}2\\3\end{bmatrix}$
3
z
3.55
2
x_1
length 1 vector
$\begin{bmatrix}0.71\\0.71\end{bmatrix}$

dot product

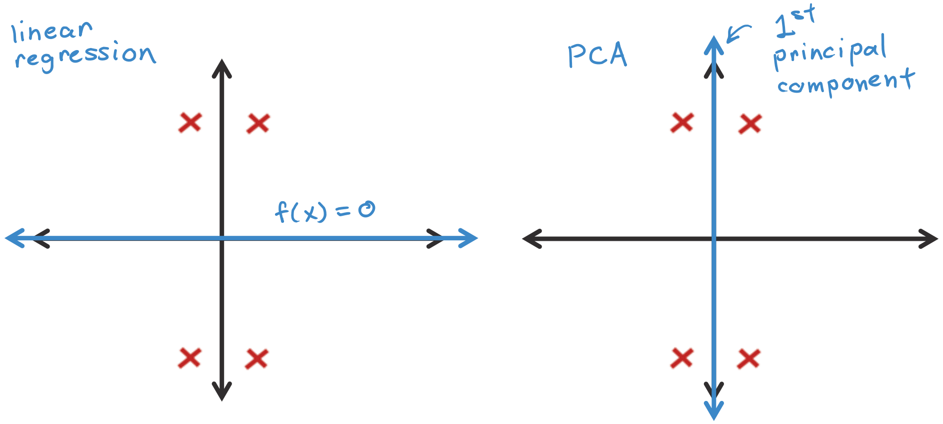
$$\begin{bmatrix}2\\3\end{bmatrix}\cdot\begin{bmatrix}0.71\\0.71\end{bmatrix}$$

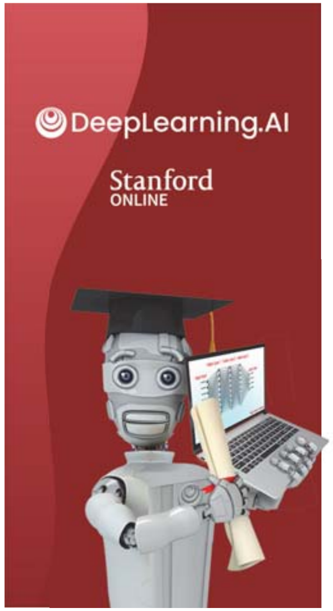
$2 \times 0.71 + 3 \times 0.71 = 3.55$

PCA is not linear regression


linear regression
(x, y)
y
minimize distance along y axis
x
PCA
$x_1, x_2, \dots, x_{50}$
find axis to retain variance (info)
x_2
z
x_1

PCA is not linear regression

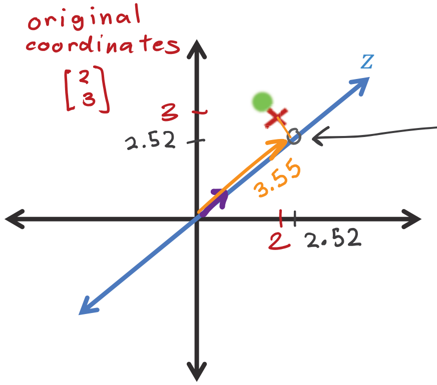




PCA

PCA in Code

Approximation to the original data



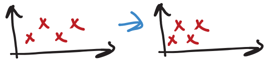
given $z = 3.55$,

find original (x_1, x_2) (approximately)

"reconstruction"

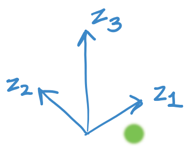
$$3.55 \times \begin{bmatrix}0.71\\0.71\end{bmatrix} = \begin{bmatrix}2.52\\2.52\end{bmatrix}$$

PCA in scikit-learn

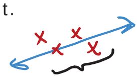


Optional pre-processing: Perform feature scaling

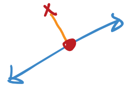
1. "fit" the data to obtain 2 (or 3) new axes (principal components) for visualization
   fit includes mean normalization



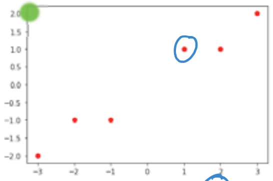
2. Optionally examine how much info variance is explained by each principal component.
   explained_variance_ratio



3. Transform (project) the data onto the new axes
   transform



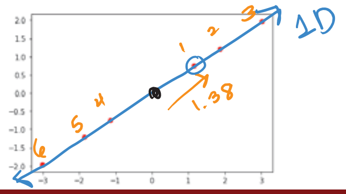
Example



```
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_   0.992
X_trans_1 = pca_1.transform(X)
X_reduced_1 = pca.inverse_transform(X_trans_1)
```



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

Applications of PCA

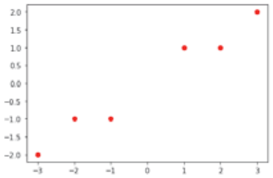
- ✩ Visualization reduce to 2 or 3 features

Less frequently used for:

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

  n = 1000 → 100

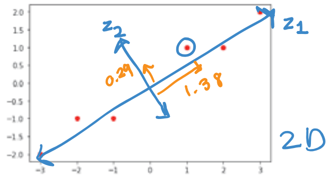
Example



```
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_   z1     z2
                                  0.992  0.008
X_trans_2 = pca.transform(X)
X_reduced_2 = pca.inverse_transform(X_trans_2)
```



```
      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]])
```