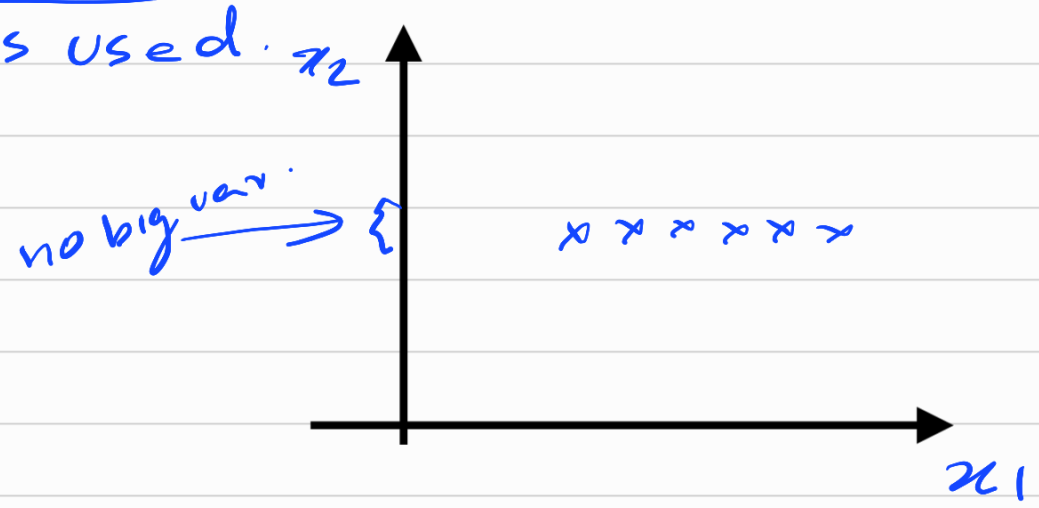
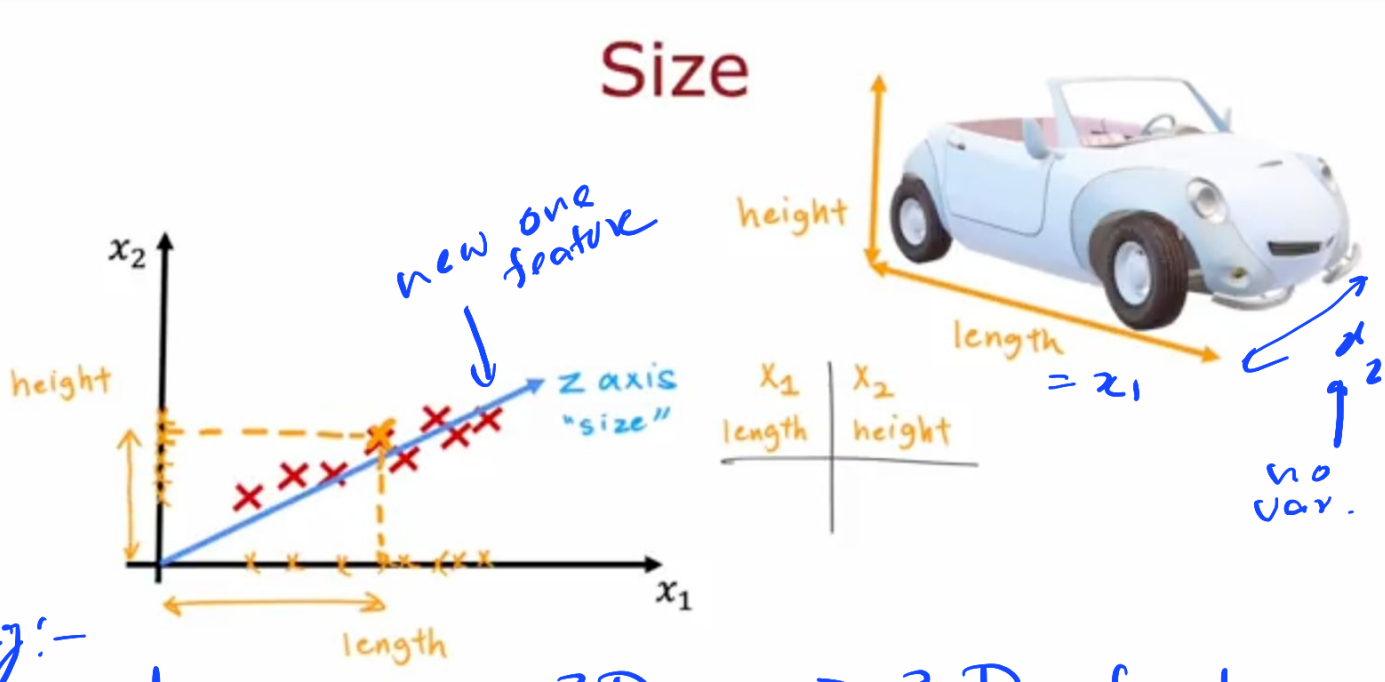
Reducing features that are less used.



or ;

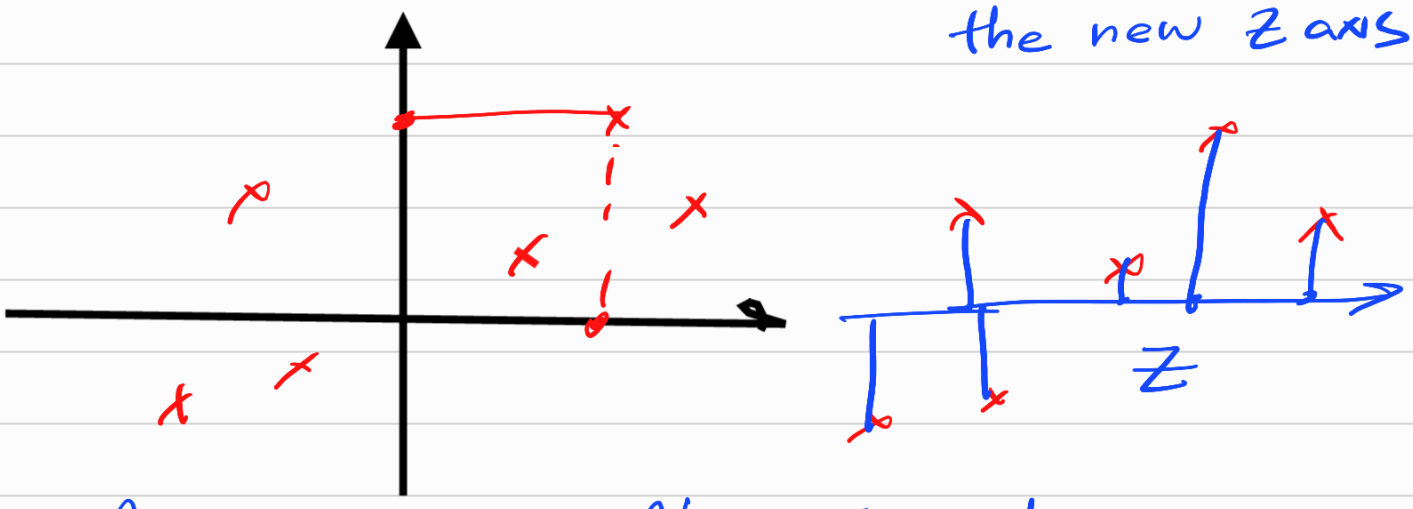


eg:- reducing 3D → 2D feature

↳ easy visualization

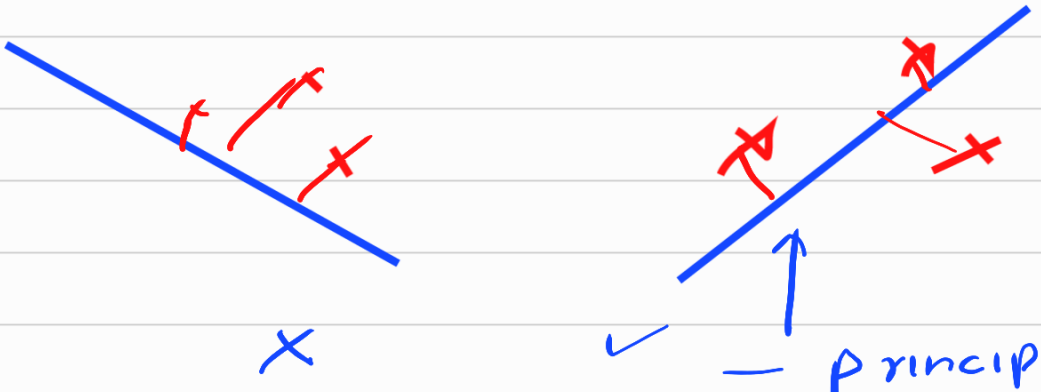
PCA Algorithm

project to the new Z axis



Preprocess → Normalize to zero mean
Scale features

If variance less ↓ — bad



X

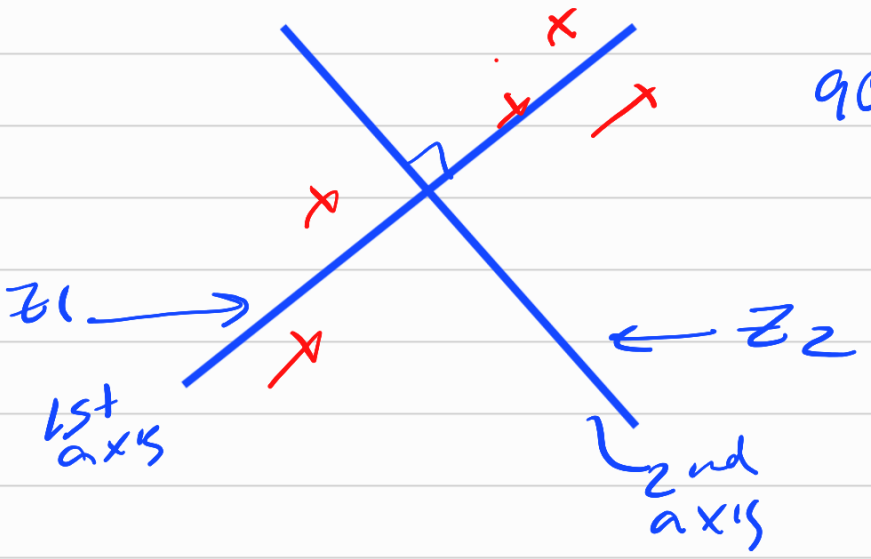
— principal component max var.
(capture more info)

(one dim → line)

Cordinate to projection — dot product

dot product

More components



90° every axis to other

In regression → labeled data
reduce variance
(dist. min between fitline)

* PCA → unsupervised
Variance ↑
to reduce no. of axis

Approximation from Z to original

→ Z = 3.55
reconstruction

multiply by the length 1 vector

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

Code ;

fit → mean normalization

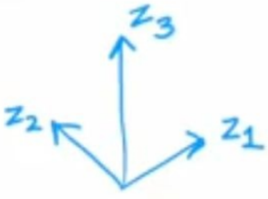
PCA in scikit-learn

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



info

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

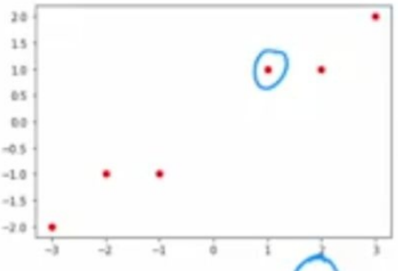
   `explained_variance_ratio`

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

   `transform`

PCA → 1 dimension

Example

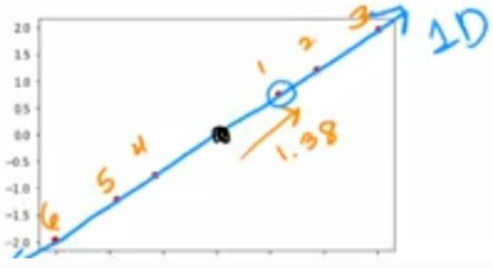


```
X = np.array([[1, 1], [2, 1], [3, 2],
              [-1, -1], [-2, -1], [-3, -2]])
```

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)
```

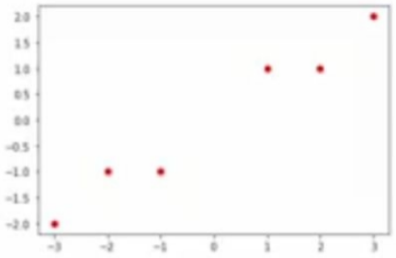
```
array([
1 [ 1.38340578],
2 [ 2.22189802],
3 [ 3.6053038 ],
4 [-1.38340578],
5 [-2.22189802],
6 [-3.6053038 ]])
```



1D

PCA → 2 dimension

Example



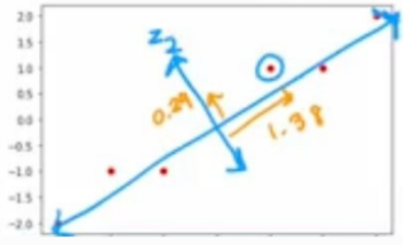
```
X = np.array([[1, 1], [2, 1], [3, 2],
              [-1, -1], [-2, -1], [-3, -2]])
```

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 z_2
0.992 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]])
```



2D

Use :— [Visualization] ← most common

Data compression

now / low } Speeding training of supervised learning model

↳ In S.V.M.