Vectors and Space

Vector

❖ Vector

- Linear algebra is a study that helps with the various calculations required for data analysis
- Analyzing data requires dealing with data consisting of a huge number of numbers
- There may be tens of thousands to hundreds of millions of numbers in a single data set, or there may be tens of thousands of such data in a single set
- Linear algebra allows complex computational processes involving large amounts of data to be described in simple, few letters

Vector

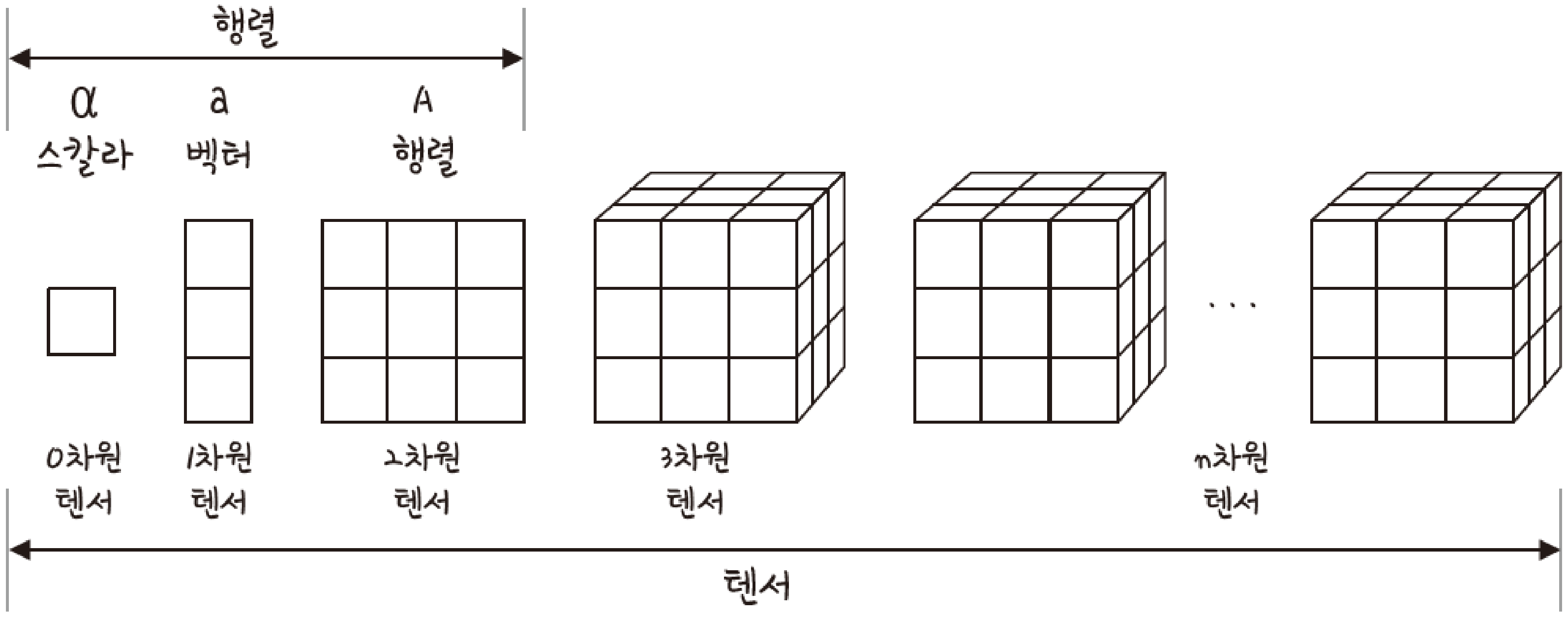
❖ Vector

- Data covered by linear algebra is largely divided into **scalar**, **vector**, **matrix**, and **tensor** types depending on the number or form
- A scalar is a **single-digit data**, and a vector is a **multi-digit data record**
- A matrix can be viewed as such a vector, i.e., **a data set with multiple data records**
- Consider of a tensor as having multiple matrices of the same size

Vector

❖ **Vector**

- In general, there is a tendency to understand scalars, vectors, and matrices on the plane as follows, and this is due to limitations in mathematical expression, so tensors should be able to consider in *n*-dimensional space



Vector

❖ Scalar

- Scalar is a single number of data
- Scala is usually in lowercase alphabetic characters, such as x, and is one of the real numbers, so in the sense that it is an element of the set R of real numbers, it is written as follows

$$x \in R$$

Vector

- **Vector**

  A point that is oriented, ordered, and exists in the space of dimensions

  - Vector means that several numbers are gathered in a specific order
  - In fact, most data records often consist of multiple numbers
  - These data bundles are called vectors in linear algebra

Vector

- **Vector**
  - In this case, the vector should be written from top to bottom in the form of one vertical line (column) and several horizontal lines (row)

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Vector

❖ **Vector**

- If the number of data that make up a vector is n, this vector is called an *n*-dimensional vector

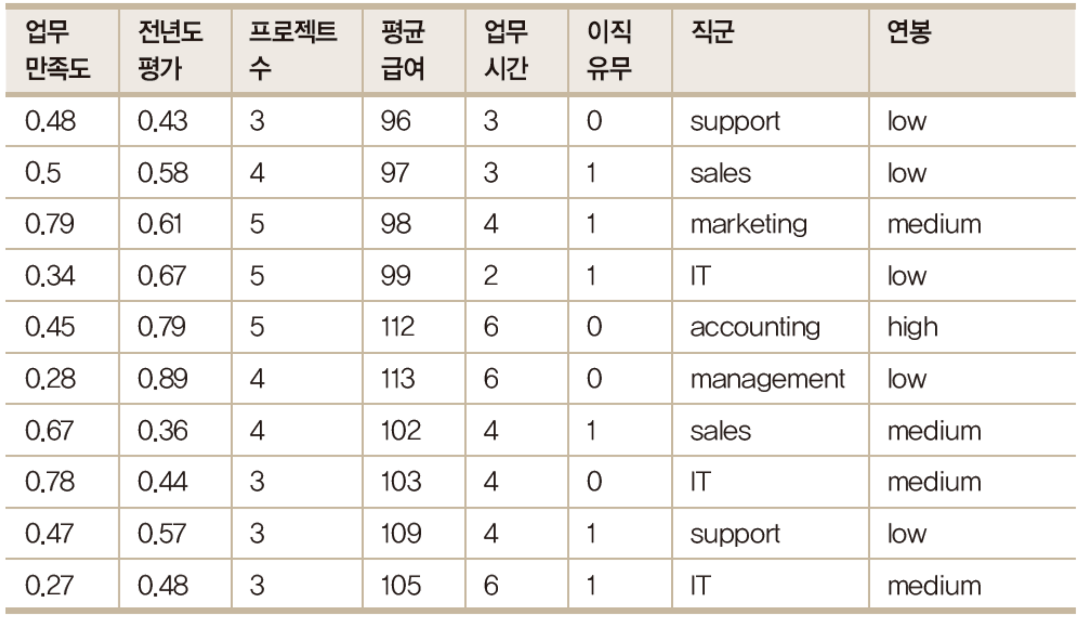
$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad x \in R^n$$

Vector

❖ **Vector**

- In fact, four items were measured: ‘업무 만족도’, ‘전년도 평가’, ‘평균 급여’, and ‘연봉’ to find out the ‘이직 유무’, as exemplified above, so it is called a four-dimensional vector

$$x \in R^4$$

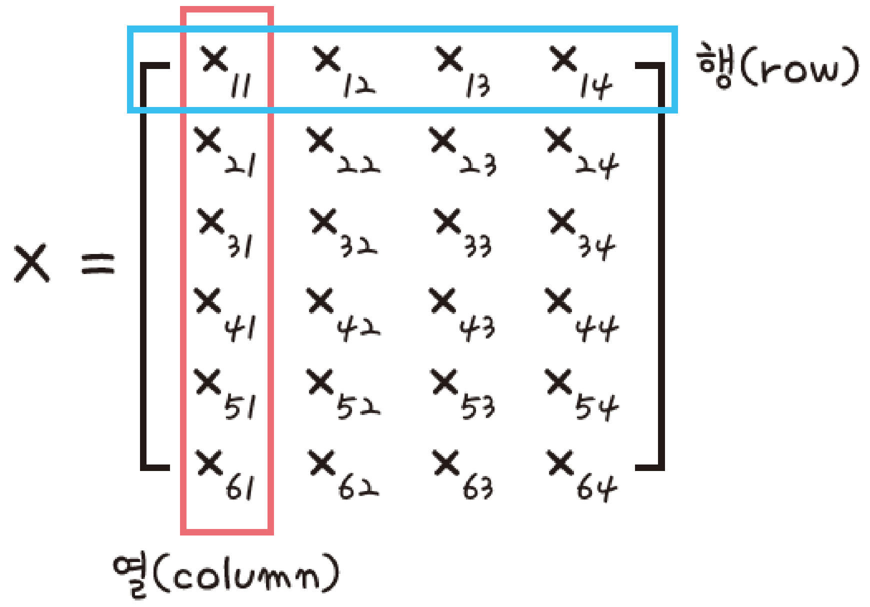


업무 만족도	전년도 평가	프로젝트 수	평균 급여	업무 시간	이직 유무	직군	연봉
0.48	0.43	3	96	3	0	support	low
0.5	0.58	4	97	3	1	sales	low
0.79	0.61	5	98	4	1	marketing	medium
0.34	0.67	5	99	2	1	IT	low
0.45	0.79	5	112	6	0	accounting	high
0.28	0.89	4	113	6	0	management	low
0.67	0.36	4	102	4	1	sales	medium
0.78	0.44	3	103	4	0	IT	medium
0.47	0.57	3	109	4	1	support	low
0.27	0.48	3	105	6	1	IT	medium

Vector

❖ **Matrix: Two-dimensional array**

- A matrix is a combination of data when there are multiple data records with multiple dimensions
- Matrix is usually capitalized with an alphabet such as X, as follows



$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \\ x_{51} & x_{52} & x_{53} & x_{54} \\ x_{61} & x_{62} & x_{63} & x_{64} \end{bmatrix}$$
행(row)
열(column)

Vector

❖ **Matrix: Two-dimensional array**

- In artificial intelligence, a row is also called a feature matrix
- When the size of this matrix is expressed in a formula, it is expressed as ‘row size x column size'
- For example, because there are six rows and four columns, we would say

$$x \in R^{6\times 4}$$

스칼라: $a \in R^{1\times 1}$

벡터: $x \in R^{4\times 1}$

Vector

- **Tensor: Array more than three dimensions**
  - Tensor means that multiple matrices of the same size are grouped together
  - A strict mathematical definition is 'mapping expressed in a multi-dimensional array', which does not mean a multi-dimensional arrangement itself
  - Multidimensional arrays are often referred to as tensors in the field of data science

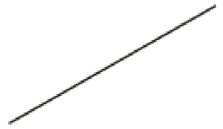
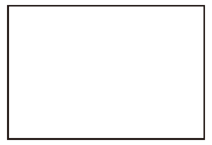
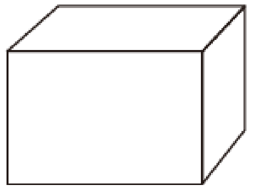
Vector

- ❖ **Tensor: Array more than three dimensions**
  - For example, as shown in Figure, a grayscale image can represent a two-dimensional matrix (arrangement) on one channel
  - On the other hand, RGB images can be expressed as two-dimensional matrices for every three R(ed), G(green), and B(lue) channels, so they can be expressed as tensors (arrays with three-dimensional values)

Channel :

The expression of color on a graphic, which has color information to construct an image, is called a channel

Vector

구분	스칼라	벡터	행렬	텐서
영문 표기	Scalar	Vector	Matrix	Tensor
표기	$x \in R$	$x \in R_n$	$x \in R_m \times n$	A
차원 (dimension)	0차원	1차원	2차원	3차원 이상
공간에서 표현				

파이썬 코드
예시
	`x = np.array`
	`(1.2)`
		`x = np.array`
		`([1, 2, 3])`
			`x = np.array`
			`([[1, 2, 3],`
			`[4, 5, 6]])`
				`x = np.array`
				`([[[[1, 2, 3],`
				`[4, 5, 6],`
				`[10, 20, 30],`
				`[200, 300, 400]]]])`

Vector

❖ Matrix

- The representation of a matrix is a rectangular arrangement of constants or variables, and it is easy to consider of the rows and columns of Excel
- The matrix are consisted of
  - Component: Each constant or variable that makes up the matrix
  - Row: Horizontal array of components
  - Column: Vertical array of ingredients
  - m×n matrix: matrix with m rows and n columns

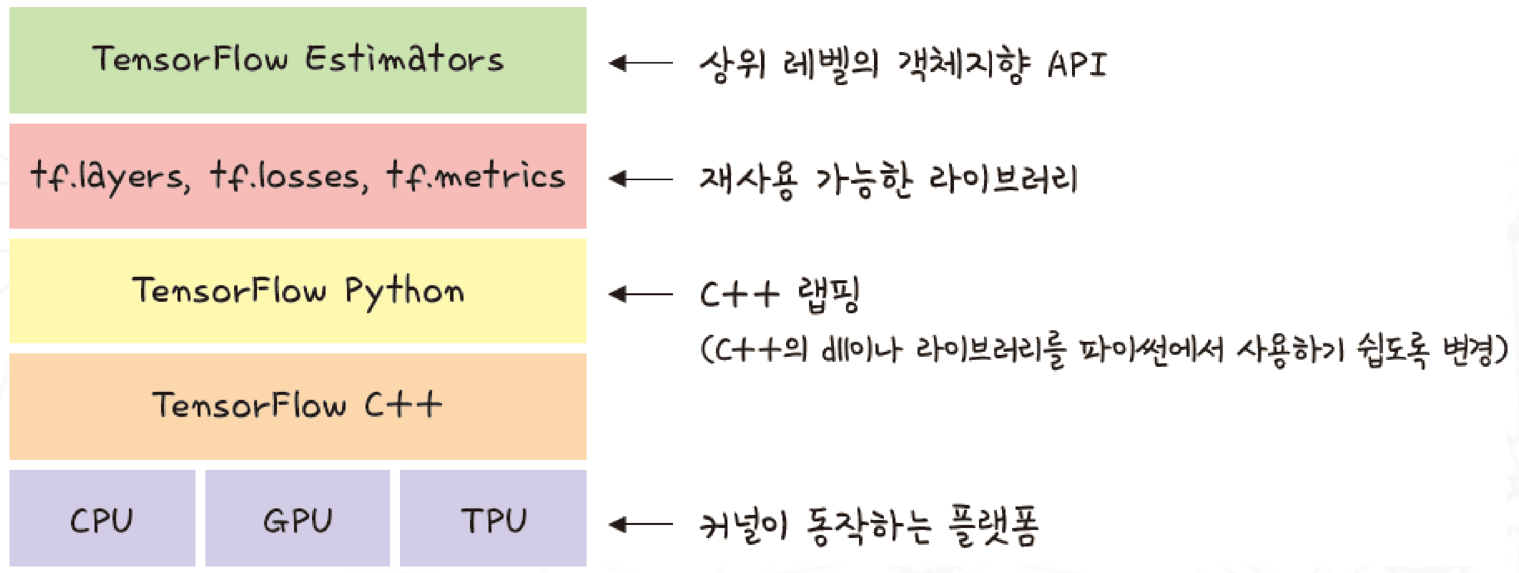
$$
a_{m\times n} = \begin{array}{c} \\ 1\text{행} \\ 2\text{행} \\ \\ m\text{행} \end{array}
\begin{array}{cccc} 1\text{열} & 2\text{열} & & n\text{열} \\ \end{array}
\left[\begin{array}{cccc} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{array}\right]
$$

a의 2행 2열

Vector

❖ **Does the tensorFlow used for data analysis mean tensor among linear algebra of data types?**

- TensorFlow is a library created by Google that offers a variety of features to make deep learning programs easy to implement
- TensorFlow itself is implemented in C++ and supports a variety of languages, including Python, Java, and Go

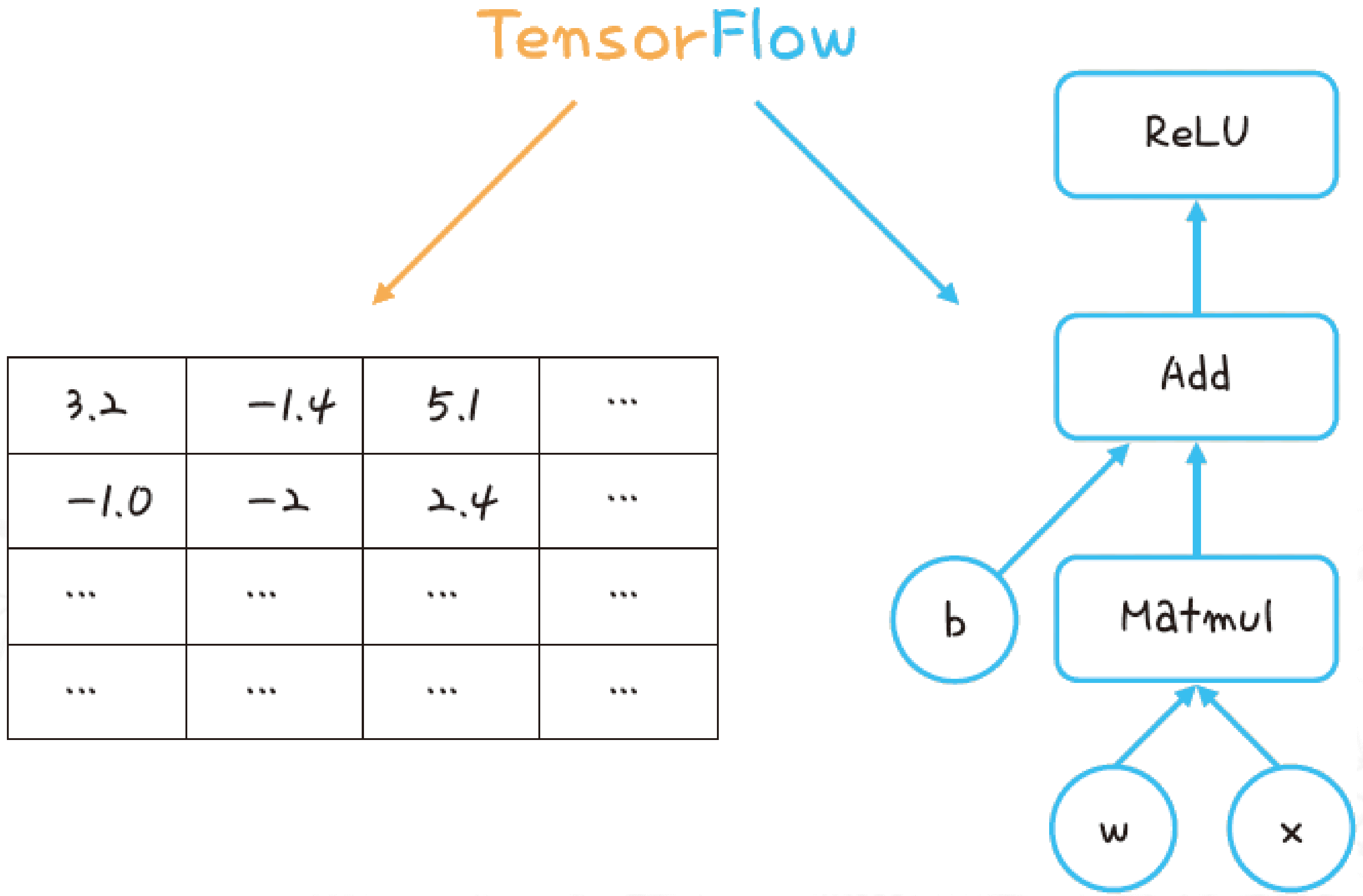


Vector

❖ **Does the tensorFlow used for data analysis mean tensor among linear algebra of data types?**

- TensorFlow calculates as a dataflow graph
- That is, data in the form of tensors flow along the graph of operations constituting the deep learning model, resulting in operations
- In deep learning, the name TensorFlow was derived by combining the tensor, which means data, and the form (Flow) in which the operation is performed along the data flow graph
- TensorFlow is currently one of the most popular deep learning libraries, offering abstraction libraries such as TensorBoard and Keras
- By providing pre-trained models, users can easily perform deep learning

Vector

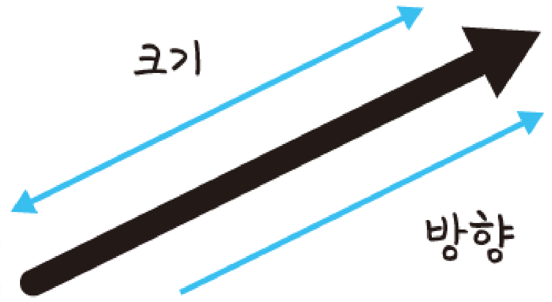


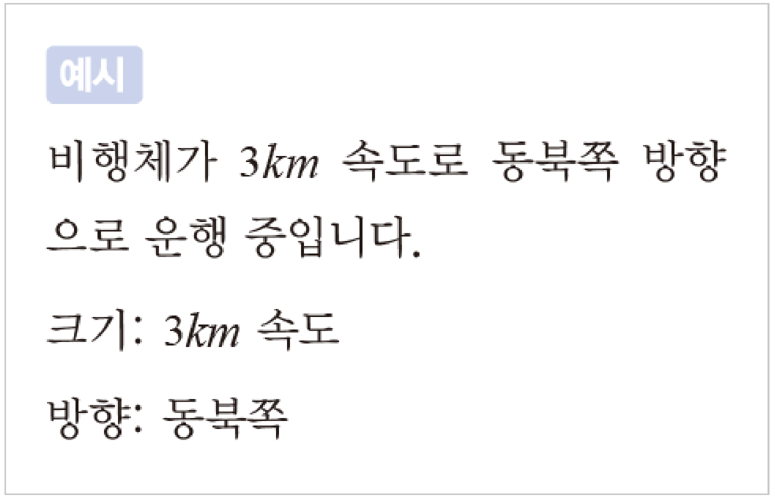
3.2	−1.4	5.1	...
−1.0	−2	2.4	...
...	...	...	...
...	...	...	...

Vector

❖ Geometric definition

- Vector is 'physical quantity with magnitude and direction' as follows
- Vectors need to be understood geometrically or visually rather than numerically because they are not associated with figures but are spatial "arrows connecting random points from random points (start points)"




예시

비행체가 $3km$ 속도로 동북쪽 방향으로 운행 중입니다.

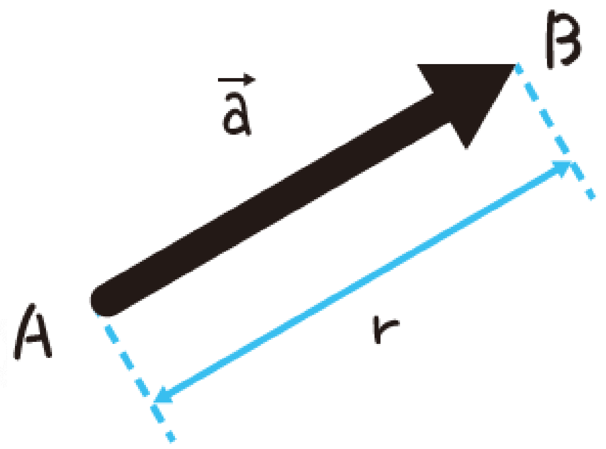
크기: $3km$ 속도

방향: 동북쪽

Vector

❖ **Characteristics of vectors**

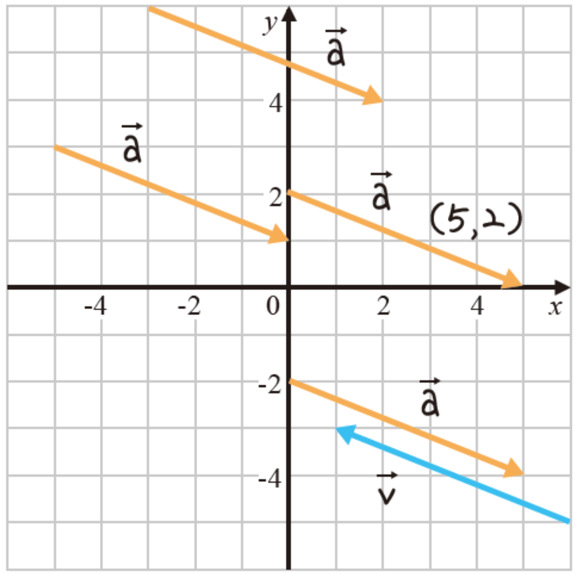
- Vectors have four main characteristics
  - (1) Directionality
    - Vectors are directional from start point (A) to end point (B), i.e. from A to B
    - The length of the line segment (r) is called the vector size
    - In other words, the vector is the addition of one more component called direction to the scalar value



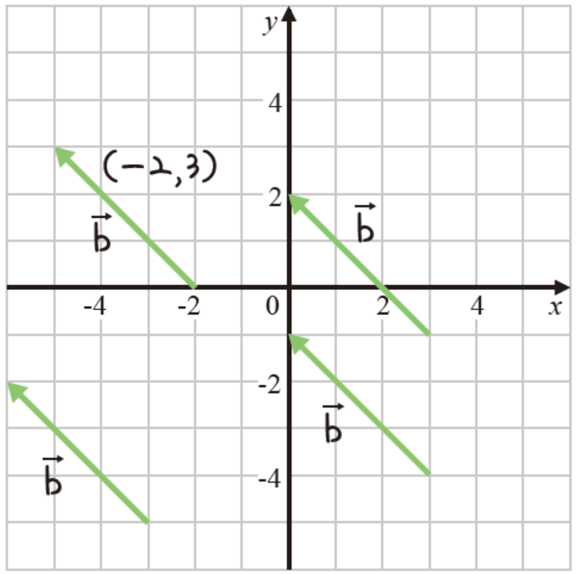
Vector

❖ Characteristics of vectors

- (2) Size and direction matter, not where it is expressed (on coordinates)
  - Same vector if size and direction are the same as follows
  - All vectors expressed in $\vec{a}$ are the same, and all vectors expressed in $\vec{b}$ are the same



$\vec{a}$는 크기와 방향이 동일한 같은 벡터



$\vec{b}$는 크기와 방향이 동일한 같은 벡터

Vector

❖ **Characteristics of vectors**

- (3) The same vector does not mean the same position on the coordinate
  - All vectors expressed in $\vec{a}$ have different positions on coordinates, but they are all the same vectors because they have the same size and direction
- (4) Same size but different direction is different vector
  - The vector $\vec{a}$ and the vector $\vec{v}$ have the same magnitude, but not the same vector because they have different directions

Vector

❖ Representation of vector

Geometric notation

- A vector is a directed line segment that connects the starting point and the end point, which is indicated by adding an arrow to the vector variable or in one character

$$\overrightarrow{AB} \text{ 혹은 } \vec{v}$$

Vector

❖ Algebraic notation

- Vectors are in lowercase bold as follows

$$\mathbf{x} = (x_1,\ x_2,\ \cdots,\ x_n) = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = [x_1 \quad x_2 \quad \cdots \quad x_n]^T$$

(T는 Transpose의 머리글자로 전치행렬을 의미합니다.

전치행렬은 열과 행을 바꾸어서 표현합니다.)

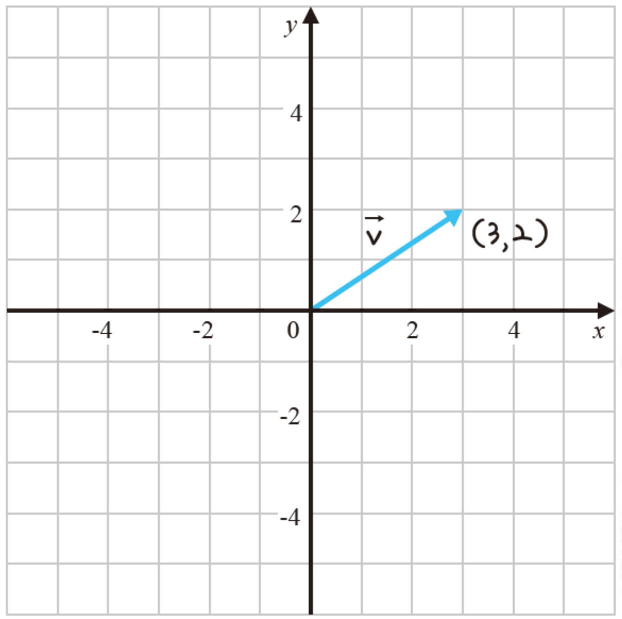
Vector

❖ **Algebraic notation**

- 3 and 2 are the components of the vector, and the $\vec{v}$ is the vector
- In particular, the previous expression (1) vector representation is called a row vector because it is expressed as a row of matrices
- (2) The vector representation is called a column vector because it is expressed as a column of a matrix

$$(1)\ \boldsymbol{x} = [3\ \ 2]$$

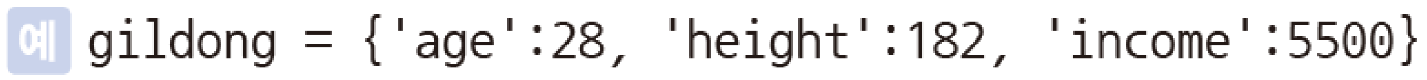
$$(2)\ \boldsymbol{x}^T = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$



Vector

❖ **Vector**

- Equations (1) and (2) above represent vectors defined in two dimensions, and can be expressed in three or more dimensions
- Vectors can represent different data as vectors
  - (1) A binary string can be represented as a vector
    - That is, the n-bit binary string 110110 may be expressed as an n-vector 1,1,0,1,0
  - (2) Can be expressed as a vector for attributes



```
예 gildong = {'age':28, 'height':182, 'income':5500}
```

Vector

❖ **Vector**

- (3) Probability distributions can be represented as vectors

```
In [4]:
{1: 1/3, 2: 2/3, 3: 0/3}
```

```
Out [4]:
{1: 0.3333333333333333, 2: 0.6666666666666666, 3: 0.0}
```

- (4) Images can be represented as vectors
  - {(i, j) | 0 ≤ i < 1024, 0 ≤ j < 768} can be viewed as a function/vector to real number R when there is a black and white image set of size 1024×768
- (5) Vectors can be used to represent not only two dimensions but also three dimensions or more of multidimensional space

Vector

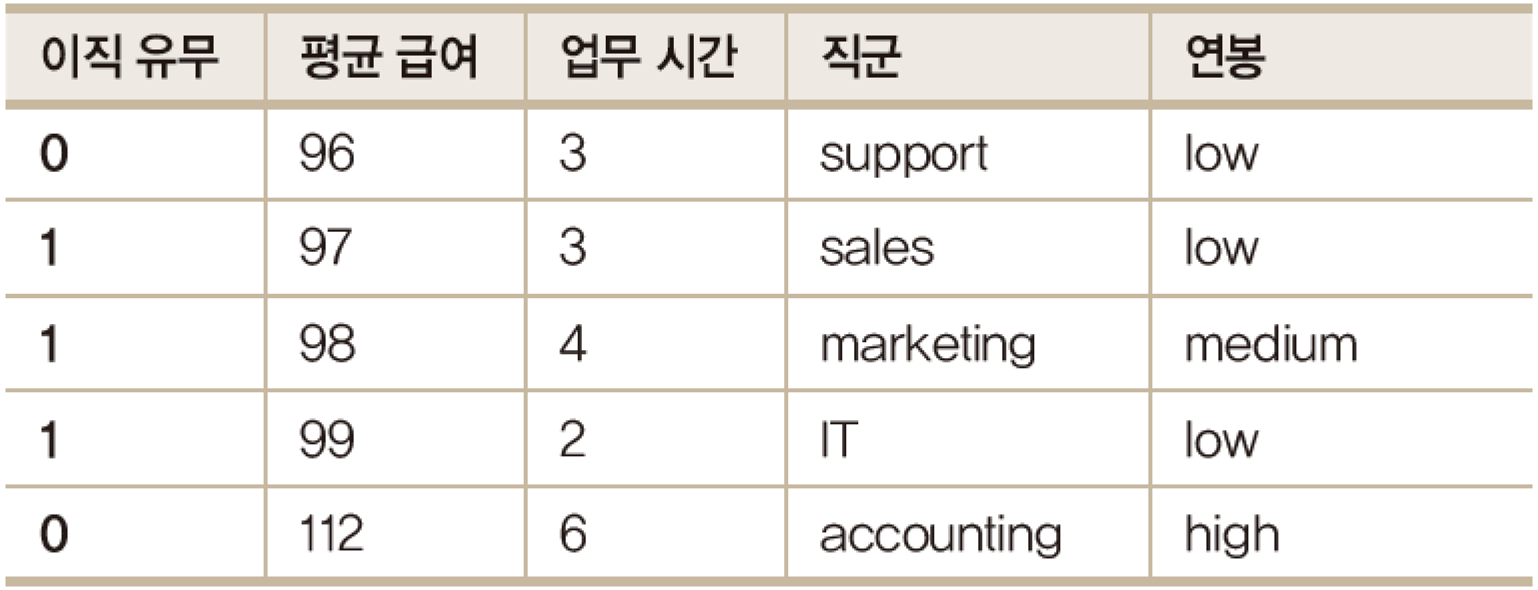
- **Why do AI use vectors**

Meaning of vector in AI

- In artificial intelligence, it is easy to understand that vectors are sets and arrays of numbers
- In artificial intelligence, the name "feature vector" is used

Vector

❖ **Why do AI use vectors**

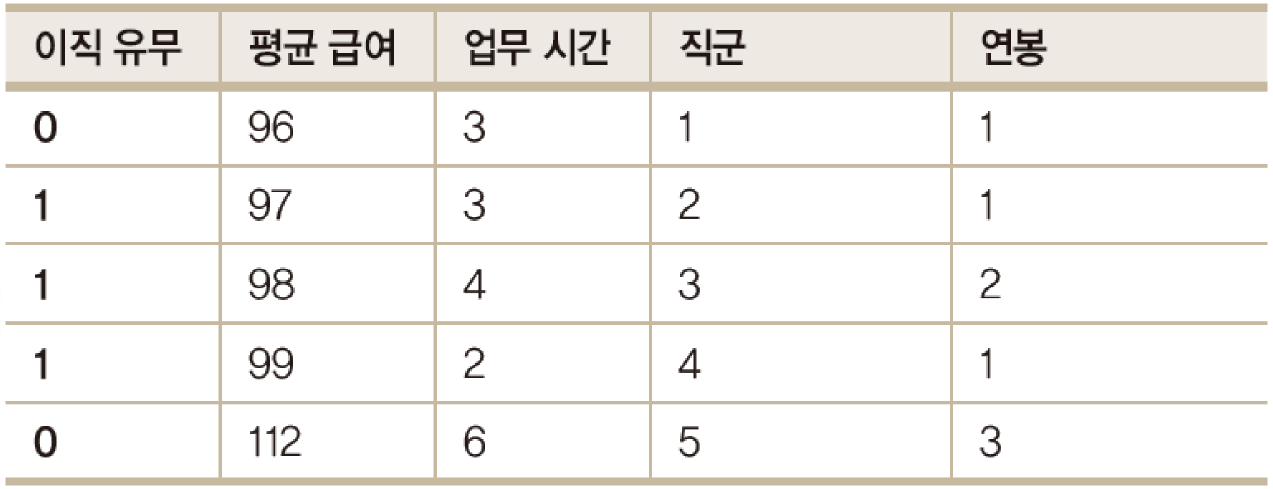


이직 유무	평균 급여	업무 시간	직군	연봉
0	96	3	support	low
1	97	3	sales	low
1	98	4	marketing	medium
1	99	2	IT	low
0	112	6	accounting	high

Vector

❖ Why do AI use vectors

- Need to convert ' 직군 ' and ' 연봉 ' into numbers
- In the characteristics of the 직군 group, ' support ' can be expressed as ' 1 ' , ' sales ' can be expressed as ' 2 ' , ' marketing ' can be expressed as ' 3 ' , ' IT ' can be expressed as ' 4 ' , and ' accounting ' can be expressed as ' 5’
- ' low ' of 연봉 as '1', 'medium' as '2', and 'high' as '3'

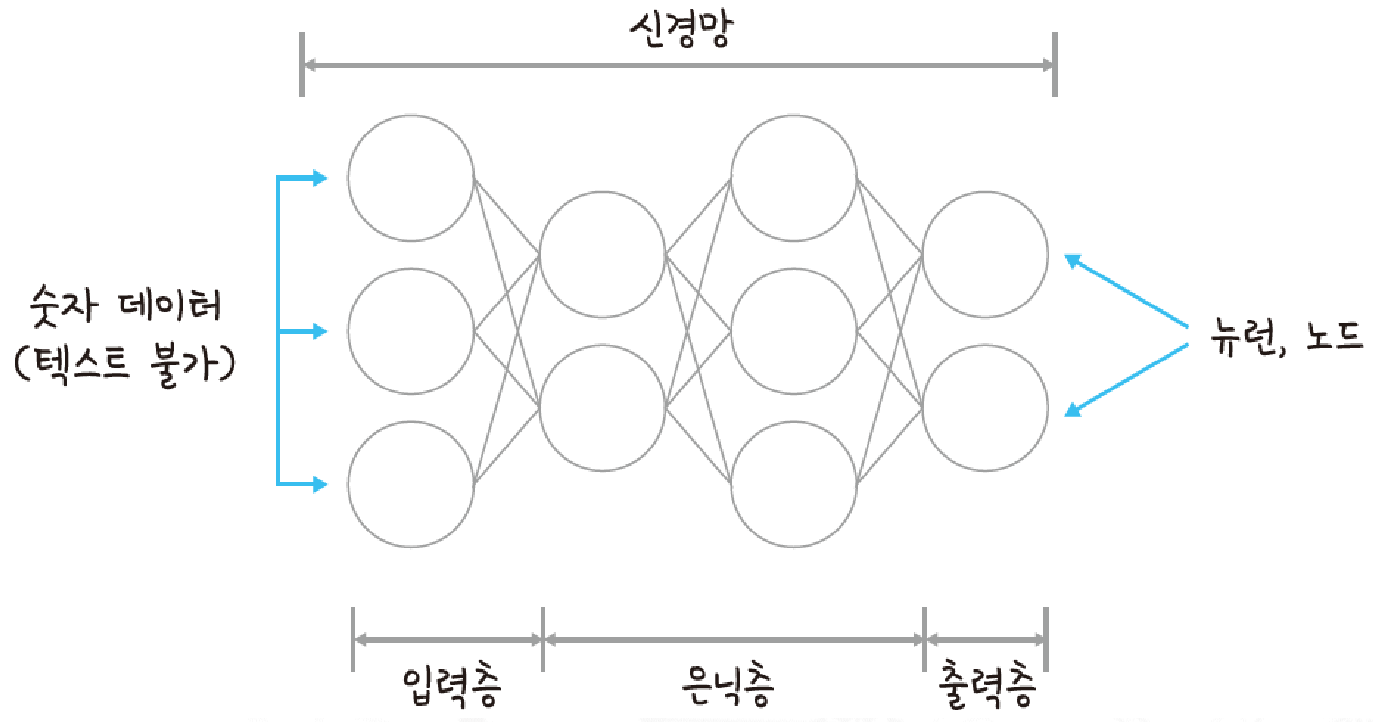


이직 유무	평균 급여	업무 시간	직군	연봉
0	96	3	1	1
1	97	3	2	1
1	98	4	3	2
1	99	2	4	1
0	112	6	5	3

Vector

❖ Why do AI use vectors

- (1) In artificial neural networks, input data can only be numeric data



Vector

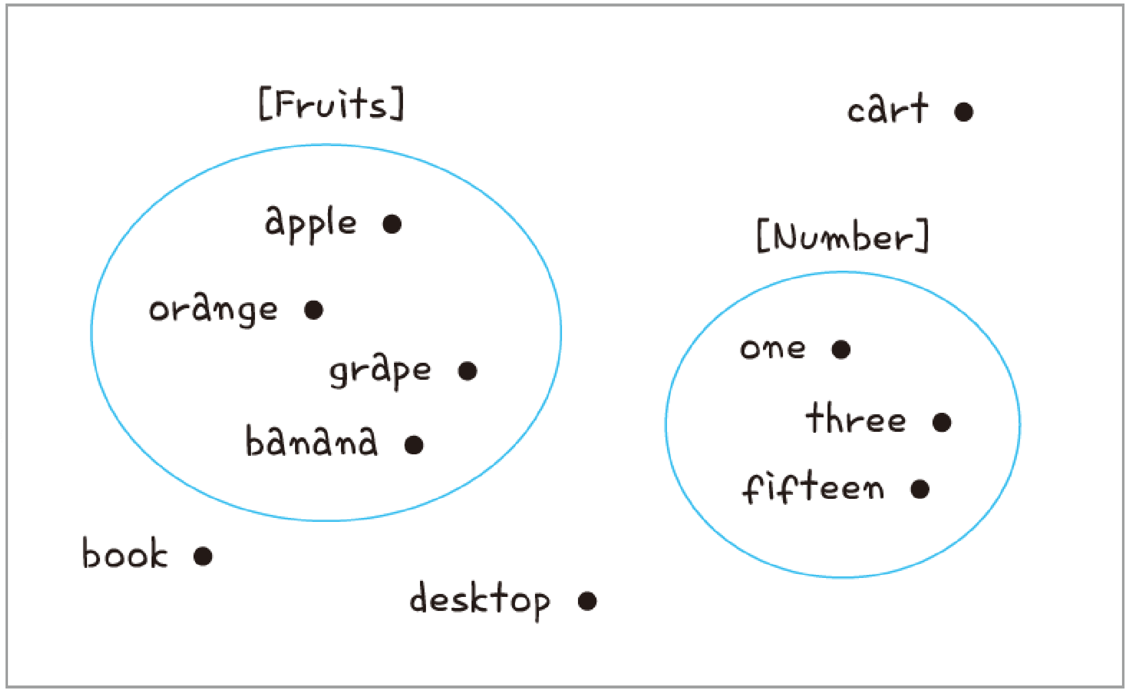
❖ **Why do AI use vectors**

- The artificial neural network of deep learning consists of three layers: an input layer, a hidden layer, and an output layer (a single layer or higher layer is also possible)
- Data input to the input node to analyze big data is only numeric data, and text data cannot be input
- Text data must be converted into numeric data and used
- However, text data should be converted into vectors by assigning correlations according to the unique meaning of a word, not by an irregular list of numbers

Vector

❖ Why do AI use vectors

- (2) Use to classify similar data
    - Let the word meaning be clustered into something similar
- Cluster: Identify the characteristics of the data and define a group of data with similar characteristics



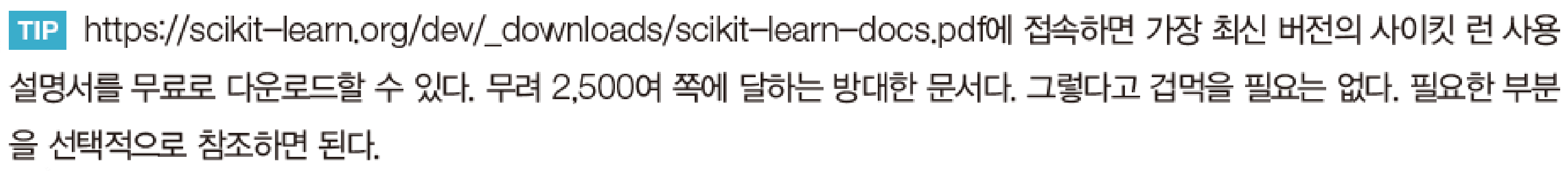
Vector

❖ Why do AI use vectors

- When the data is composed of a total of ten, words with similar word meanings can be classified and vectorized
- {apple, orange, grape, banana} can be clustered into fruits, so it is vectorized by setting it to the number '1'
- {one, three, and ten} can be clustered by a number (Number), so it is vectorized by setting it to the number '2'
- In other words, rather than giving a random number, the number is applied in consideration of meaning and converted into a vector
- To do artificial intelligence, you need to find out what characteristics the data has and make it a vector
- In other words, making data a vector is the beginning of artificial intelligence

Examples

- ❖ Install the scikit-learn library
  - pip install scikit-learn



TIP https://scikit-learn.org/dev/_downloads/scikit-learn-docs.pdf에 접속하면 가장 최신 버전의 사이킷 런 사용 설명서를 무료로 다운로드할 수 있다. 무려 2,500여 쪽에 달하는 방대한 문서다. 그렇다고 겁먹을 필요는 없다. 필요한 부분을 선택적으로 참조하면 된다.

Load ‘iris’ Dataset

❖ Load the dataset

프로그램 3-1(a) iris 데이터셋 읽기

```
01  from sklearn import datasets
02
03  d=datasets.load_iris()    # iris 데이터셋을 읽고
04  print(d.DESCR)            # 내용을 출력
```

- 01행: sklearn 모듈의 datasets 클래스를 불러옴
- 03행: load_iris 함수를 호출해 iris 데이터셋을 읽어 객체 d에 저장
- 04행: 객체 d의 DESCR 변수를 출력

❖ Terminology

- Dataset
- Feature vector
- Class

TIP 기계 학습이 사용하는 데이터는 여러 개의 샘플을 담고 있어서 데이터셋(data set)이라 부르기도 한다. 이 책에서는 데이터와 데이터셋을 엄밀히 구분하지 않고 함께 사용하는데, 데이터셋은 iris처럼 특정한 데이터를 가리킬 때 주로 사용한다.

Load ‘iris’ Dataset

```
Iris plants dataset
--------------------

**Data Set Characteristics:**

    :Number of Instances: 150 (50 in each of three classes)
    :Number of Attributes: 4 numeric, predictive attributes and the class
    :Attribute Information:
        - sepal length in cm
        - sepal width in cm
        - petal length in cm
        - petal width in cm
        - class:
                - Iris-Setosa
                - Iris-Versicolour
                - Iris-Virginica

    :Summary Statistics:

    ============== ==== ==== ======= ===== ====================
                    Min  Max   Mean    SD   Class Correlation
    ============== ==== ==== ======= ===== ====================
    sepal length:   4.3  7.9   5.84   0.83    0.7826
    sepal width:    2.0  4.4   3.05   0.43   -0.4194
    petal length:   1.0  6.9   3.76   1.76    0.9490  (high!)
    petal width:    0.1  2.5   1.20   0.76    0.9565  (high!)
    ============== ==== ==== ======= ===== ====================
:Missing Attribute Values: None
    :Class Distribution: 33.3% for each of 3 classes.
    :Creator: R.A. Fisher
    :Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
    :Date: July, 1988
...
```

150개의 샘플

네 개의 특징(feature)

세 개의 부류

Load ‘iris’ Dataset

❖ ‘iris’ dataset

프로그램 3-1(b) iris의 내용 살펴보기

```
for i in range(0,len(d.data)):        # 샘플을 순서대로 출력
    print(i+1,d.data[i],d.target[i])
```

```
1 [5.1 3.5 1.4 0.2] 0
2 [4.9 3.  1.4 0.2] 0
3 [4.7 3.2 1.3 0.2] 0
4 [4.6 3.1 1.5 0.2] 0
...

51 [7.  3.2 4.7 1.4] 1
52 [6.4 3.2 4.5 1.5] 1
53 [6.9 3.1 4.9 1.5] 1
54 [5.5 2.3 4.  1.3] 1
...
101 [6.3 3.3 6.  2.5] 2
102 [5.8 2.7 5.1 1.9] 2
103 [7.1 3.  5.9 2.1] 2
104 [6.3 2.9 5.6 1.8] 2
...
```

d.target(레이블)

d.data(특징 벡터)

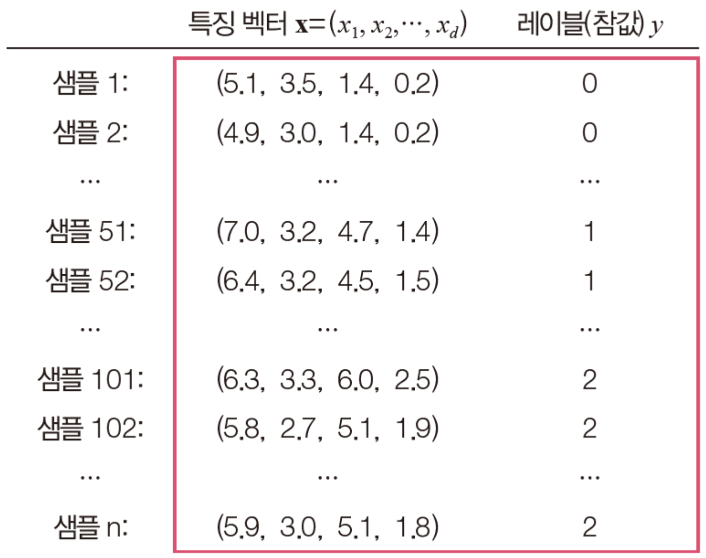
Representation of dataset

❖ Representing samples as feature vectors and labels

- Feature vectors are denoted by x

특징 벡터: $\mathbf{x}=(x_1,x_2,\cdots,x_d)$

  - d is the number of features called the dimension of the feature vector
- Labels are 0,1,2,...A value of ,c-1 or 1,2,...A value of ,c-1,c or one hot code
  - One hot code is a binary sequence with only one element
  - Ex) Setosa: (1,0,0), Versicolor: (0,1,0), Virginica: (0,0,1)



	특징 벡터 $\mathbf{x}=(x_1, x_2,\cdots, x_d)$	레이블(참값) y
샘플 1:	(5.1, 3.5, 1.4, 0.2)	0
샘플 2:	(4.9, 3.0, 1.4, 0.2)	0
…	…	…
샘플 51:	(7.0, 3.2, 4.7, 1.4)	1
샘플 52:	(6.4, 3.2, 4.5, 1.5)	1
…	…	…
샘플 101:	(6.3, 3.3, 6.0, 2.5)	2
샘플 102:	(5.8, 2.7, 5.1, 1.9)	2
…	…	…
샘플 n:	(5.9, 3.0, 5.1, 1.8)	2

iris 데이터셋
(n=150, d=4)

Data Distribution of Feature Space

❖ iris dataset

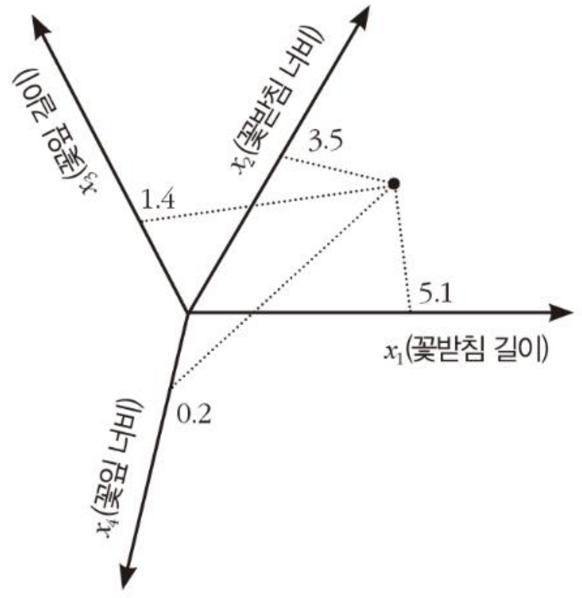
- Distribution of data in a three-dimensional space, excluding one data dimension

프로그램 3-2 iris 데이터의 분포를 특징 공간에 그리기

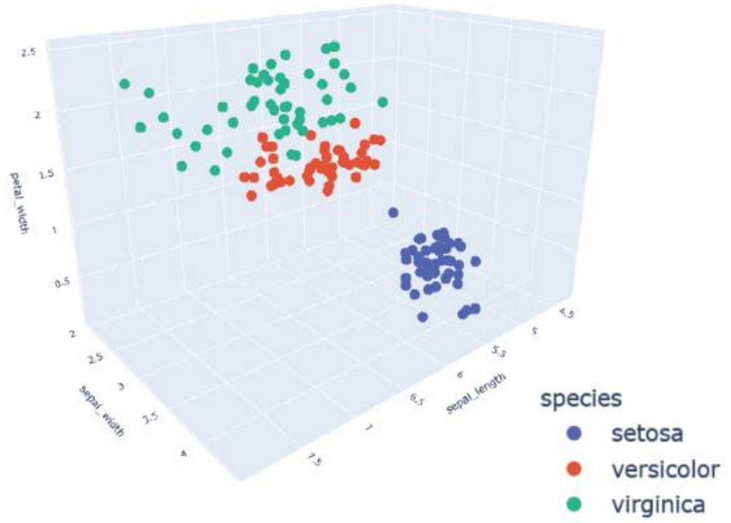
```python
import plotly.express as px

df = px.data.iris()
fig = px.scatter_3d(df, x='sepal_length', y='sepal_width', z='petal_width',
color='species')        # petal_length를 제외하여 3차원 공간 구성
fig.show(renderer="browser")
```

Data Distribution of Feature Space

- ❖ Observe the distribution of data in the feature space
  - ▪ Setosa is distributed downward and Virginica is distributed upward for the vertical width
    - • Petal width is excellent in discernment
  - ▪ The segmental width axis overlaps a lot in three categories, so it is less sensible
  - ▪ As a whole, the three classes occupy different areas of the three-dimensional space, with several samples overlapping

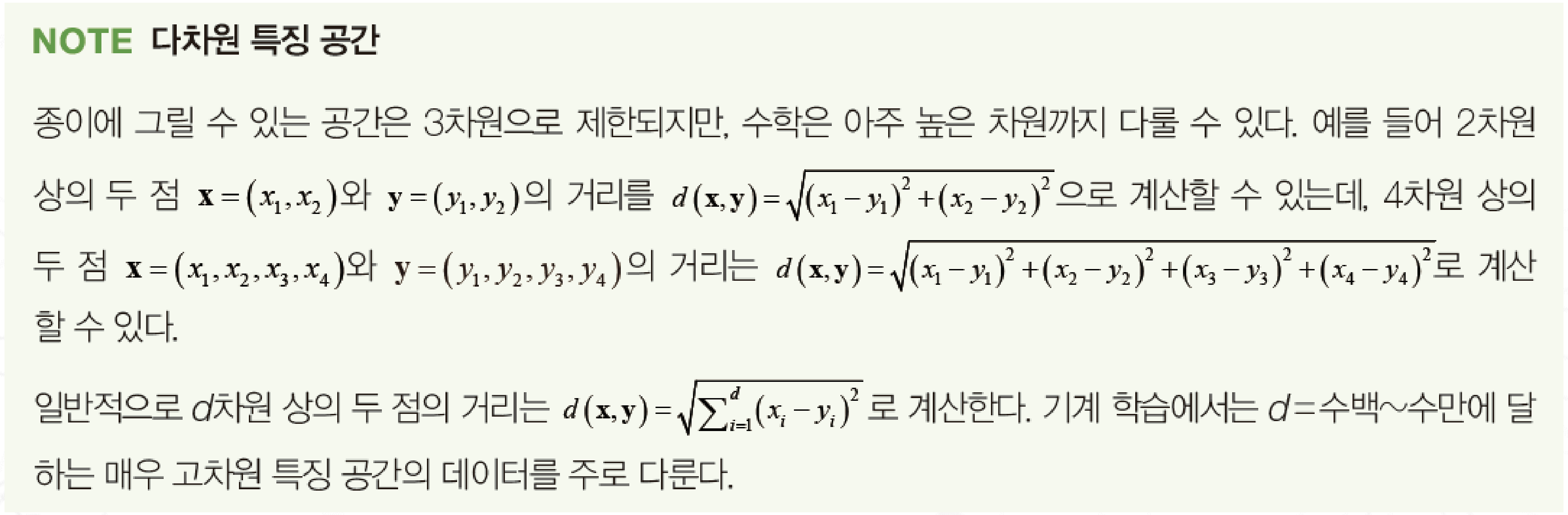


(a) 4차원 특징 공간(가상의 그림)



(b) 꽃잎 길이 축을 제외한 3차원 특징 공간

Data Distribution of Feature Space



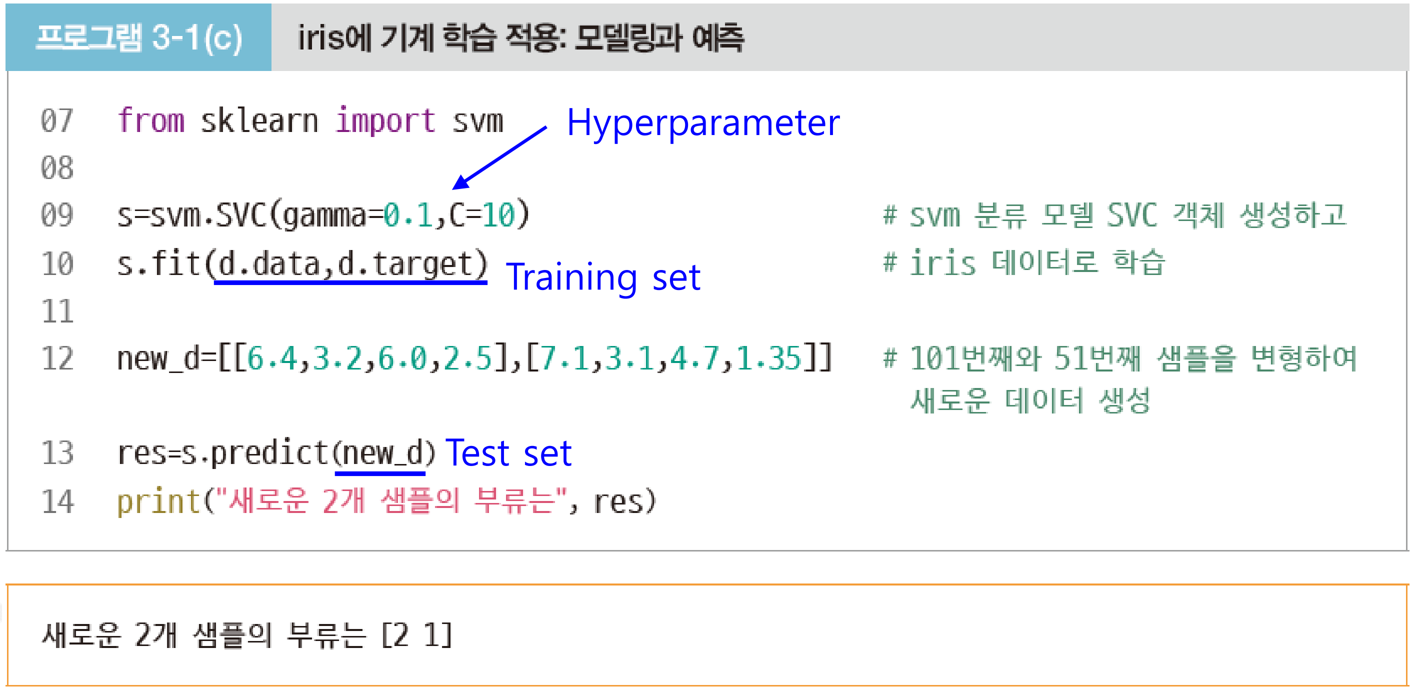
NOTE 다차원 특징 공간

종이에 그릴 수 있는 공간은 3차원으로 제한되지만, 수학은 아주 높은 차원까지 다룰 수 있다. 예를 들어 2차원 상의 두 점 $\mathbf{x}=(x_1,x_2)$와 $\mathbf{y}=(y_1,y_2)$의 거리를 $d(\mathbf{x},\mathbf{y})=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2}$으로 계산할 수 있는데, 4차원 상의 두 점 $\mathbf{x}=(x_1,x_2,x_3,x_4)$와 $\mathbf{y}=(y_1,y_2,y_3,y_4)$의 거리는 $d(\mathbf{x},\mathbf{y})=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2+(x_3-y_3)^2+(x_4-y_4)^2}$로 계산할 수 있다.

일반적으로 d차원 상의 두 점의 거리는 $d(\mathbf{x},\mathbf{y})=\sqrt{\sum_{i=1}^{d}(x_i-y_i)^2}$ 로 계산한다. 기계 학습에서는 d=수백∼수만에 달하는 매우 고차원 특징 공간의 데이터를 주로 다룬다.

Modeling and Prediction

❖ Using the support vector machine model



프로그램 3-1(c) iris에 기계 학습 적용: 모델링과 예측

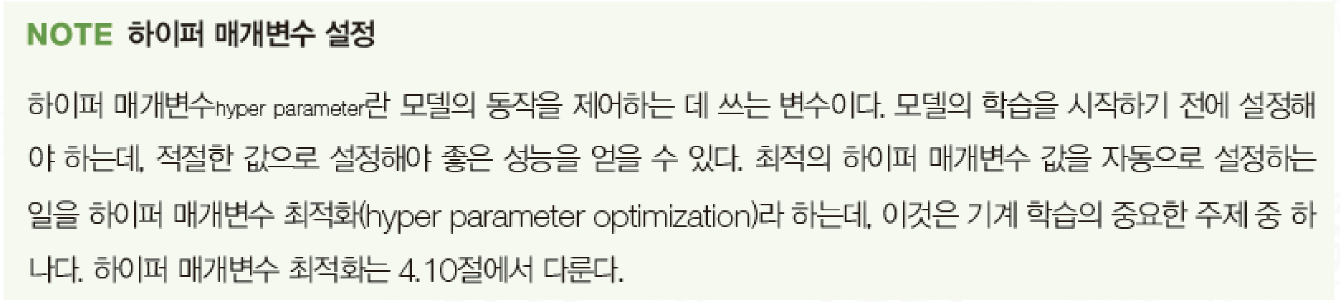
```
07  from sklearn import svm
08
09  s=svm.SVC(gamma=0.1,C=10)                    # svm 분류 모델 SVC 객체 생성하고
10  s.fit(d.data,d.target)                       # iris 데이터로 학습
11
12  new_d=[[6.4,3.2,6.0,2.5],[7.1,3.1,4.7,1.35]] # 101번째와 51번째 샘플을 변형하여
                                                   새로운 데이터 생성
13  res=s.predict(new_d)
14  print("새로운 2개 샘플의 부류는", res)
```

```
새로운 2개 샘플의 부류는 [2 1]
```

- 09행: SVM의 분류기 모델 SVC 클래스의 객체를 생성하여 s에 저장
- 10행: 객체 s의 fit 함수는 훈련 집합을 가지고 학습을 수행
  (매개변수로 특징 벡터 iris.data와 레이블 iris,target을 설정)
- 13행: 객체 s의 predict 함수는 테스트 집합을 가지고 예측 수행

Divide into Training/Validation/Test

- Training/Validation/Test
  - Training set
    - Data used to learn machine learning models that provide both feature vector and label information
  - Test Set
    - Data used to measure the performance of a learned model, which provides only feature vector information when predicting, and uses label information when measuring accuracy with prediction results

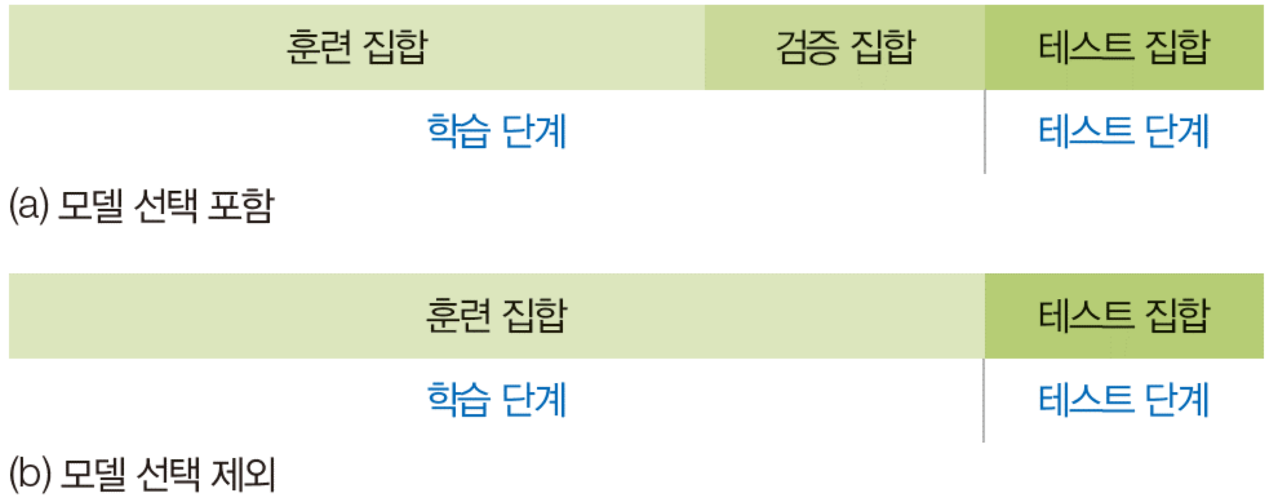


NOTE 하이퍼 매개변수 설정

하이퍼 매개변수hyper parameter란 모델의 동작을 제어하는 데 쓰는 변수이다. 모델의 학습을 시작하기 전에 설정해야 하는데, 적절한 값으로 설정해야 좋은 성능을 얻을 수 있다. 최적의 하이퍼 매개변수 값을 자동으로 설정하는 일을 하이퍼 매개변수 최적화(hyper parameter optimization)라 하는데, 이것은 기계 학습의 중요한 주제 중 하나다. 하이퍼 매개변수 최적화는 4.10절에서 다룬다.

Divide into Training/Validation/Test

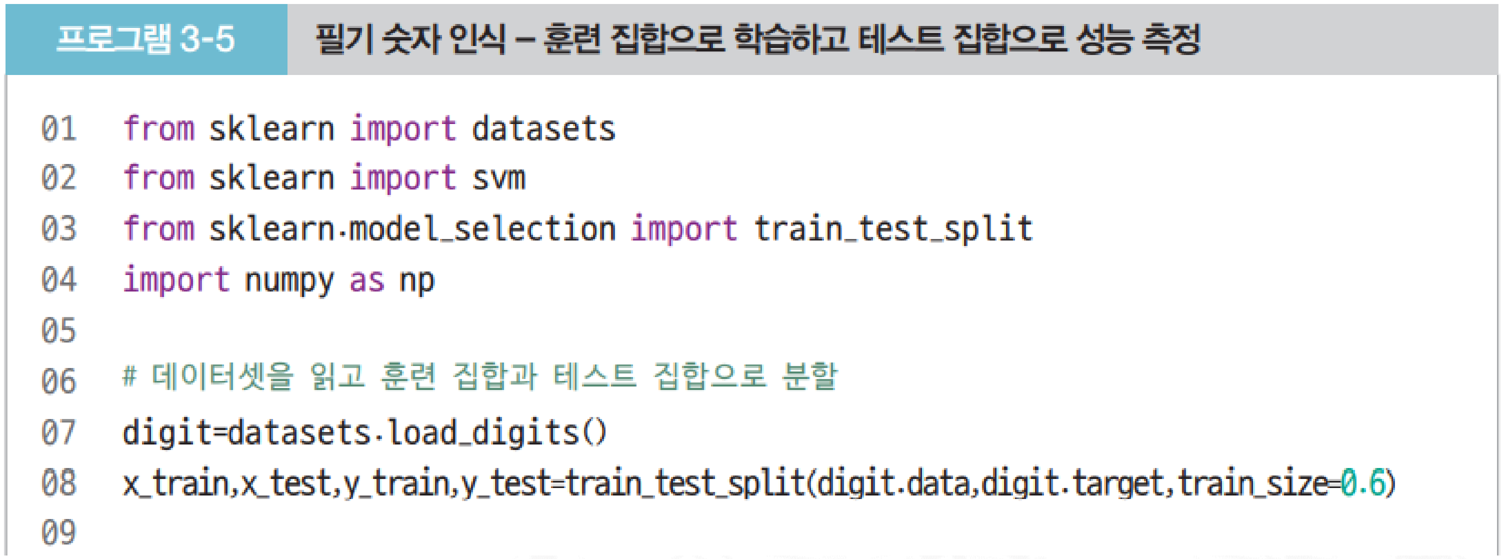
- ❖ Divide the given data into training, validation, and test sets at an appropriate rate
  - Model selection included: divided into training/validation/test sets
  - Exclude model selection: split into training/test sets



Divide into Training/Validation/Test

❖ Exclude the model selection

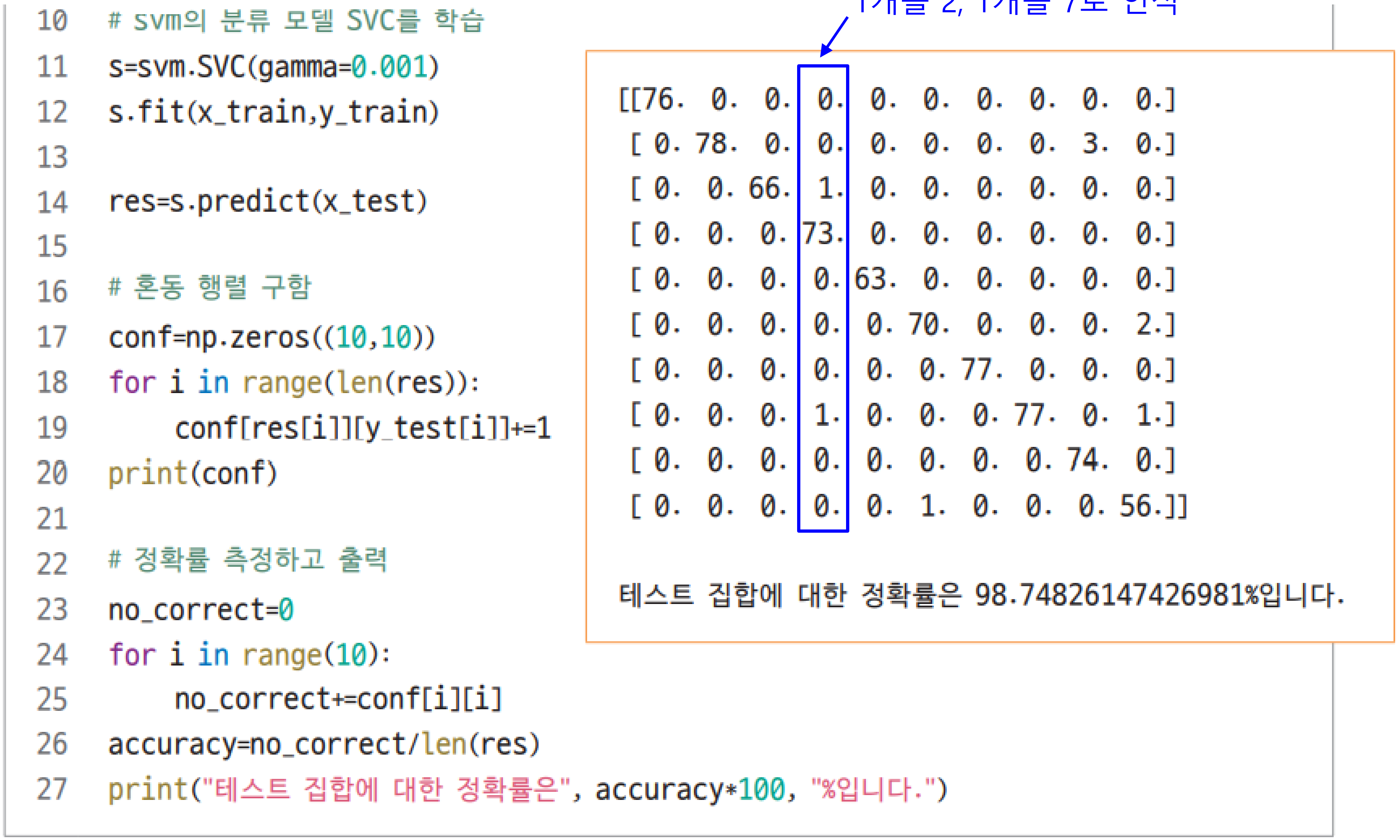
- 08행: train_test_split 함수로 훈련 60%, 테스트 40%로 랜덤 분할
- 12행: 훈련 집합 x_train, y_train을 fit 함수에 주어 학습 수행
- 14행: 테스트 집합의 특징 벡터 x_test를 predict 함수에 주어 예측 수행
- 17~20행: 테스트 집합의 레이블 y_test를 가지고 혼동 행렬 계산



프로그램 3-5 필기 숫자 인식 – 훈련 집합으로 학습하고 테스트 집합으로 성능 측정

```
01  from sklearn import datasets
02  from sklearn import svm
03  from sklearn.model_selection import train_test_split
04  import numpy as np
05
06  # 데이터셋을 읽고 훈련 집합과 테스트 집합으로 분할
07  digit=datasets.load_digits()
08  x_train,x_test,y_train,y_test=train_test_split(digit.data,digit.target,train_size=0.6)
09
```

Divide into Training/Validation/Test



```python
# svm의 분류 모델 SVC를 학습
s=svm.SVC(gamma=0.001)
s.fit(x_train,y_train)

res=s.predict(x_test)

# 혼동 행렬 구함
conf=np.zeros((10,10))
for i in range(len(res)):
    conf[res[i]][y_test[i]]+=1
print(conf)

# 정확률 측정하고 출력
no_correct=0
for i in range(10):
    no_correct+=conf[i][i]
accuracy=no_correct/len(res)
print("테스트 집합에 대한 정확률은", accuracy*100, "%입니다.")
```

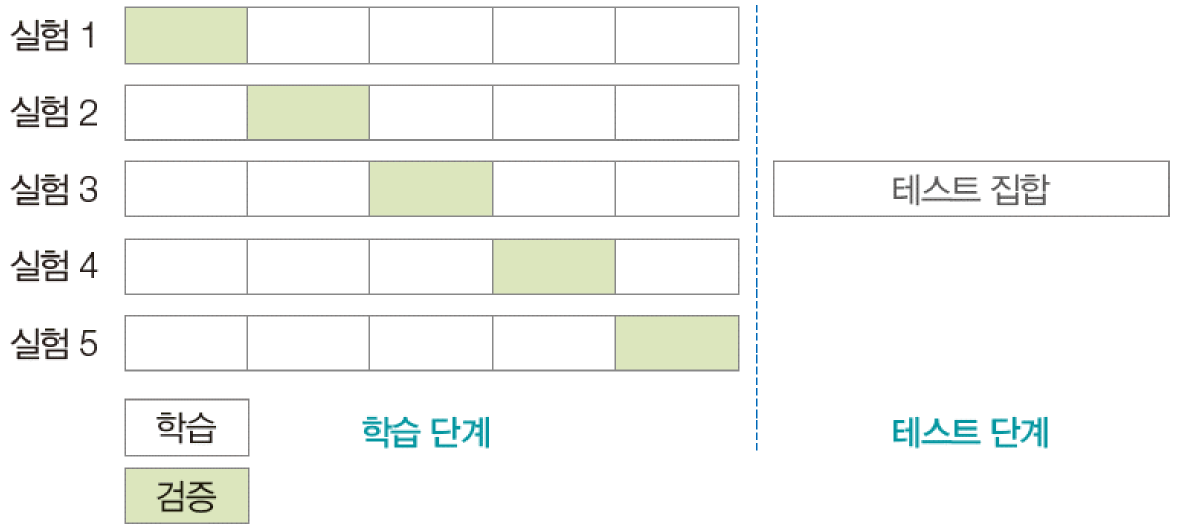
예) 부류 3에 속하는 75개 샘플 중 73개를 3, 1개를 2, 1개를 7로 인식

```
[[76.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0. 78.  0.  0.  0.  0.  0.  0.  3.  0.]
 [ 0.  0. 66.  1.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0. 73.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0. 63.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0. 70.  0.  0.  0.  2.]
 [ 0.  0.  0.  0.  0.  0. 77.  0.  0.  0.]
 [ 0.  0.  0.  1.  0.  0.  0. 77.  0.  1.]
 [ 0.  0.  0.  0.  0.  0.  0.  0. 74.  0.]
 [ 0.  0.  0.  0.  0.  1.  0.  0.  0. 56.]]

테스트 집합에 대한 정확률은 98.74826147426981%입니다.
```

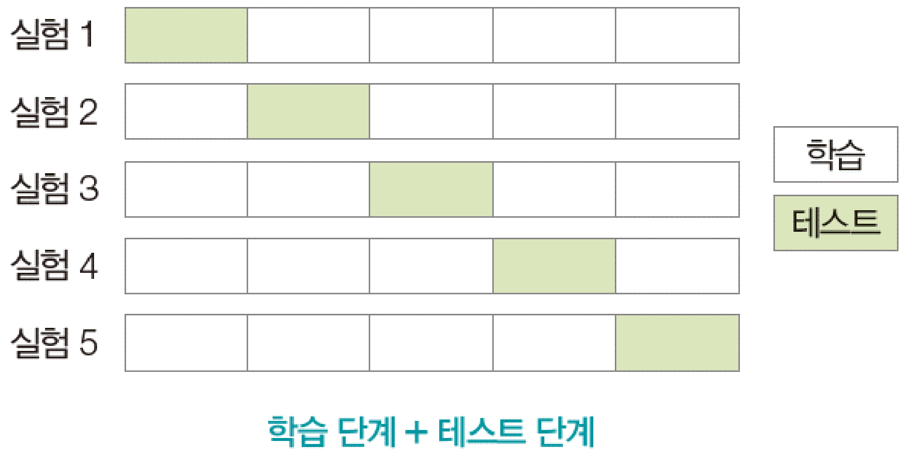
Cross-Validation

- Limitations of training/test set division
  - Likelihood of accidental high or accidental low accuracy
- k-fold cross validation
  - Use the training set divided into k subsets
  - Measure performance by learning with k-1 leaving one and then leaving it
  - Increase reliability by averaging k performance

Cross-Validation



(a) 모델 선택 포함



(b) 모델 선택 제외

Cross-Validation

프로그램 3-6 필기 숫자 인식 – 교차 검증으로 성능 측정

```python
from sklearn import datasets
from sklearn import svm
from sklearn.model_selection import cross_val_score
import numpy as np

digit=datasets.load_digits()
s=svm.SVC(gamma=0.001)
accuracies=cross_val_score(s,digit.data,digit.target,cv=5) # 5-겹 교차 검증

print(accuracies)
print("정확률(평균)=%0.3f, 표준편차=%0.3f"%(accuracies.mean()*100,accuracies.std()))
```

```
[0.97527473 0.95027624 0.98328691 0.99159664 0.95774648]
정확률(평균)=97.164, 표준편차=0.015
```

Linear combination and linearly independent

❖ Linear combination

- Linear combination is an operation that combines the scalar multiplication and addition of a vector to obtain a new vector
- From a geometric point of view, the scalar-vector product reduces or increases the length of the vector, and the addition of the two vectors coincides with the diagonal of the parallelogram formed by the two vectors

Linear combination and linearly independent

❖ **Linear combination**

- In other words, when there are two vectors $\vec{a}=\begin{bmatrix} a_1 \\ a_2 \end{bmatrix}, \vec{b}=\begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$

the operation of multiplying by random real numbers (C1, C2) to derive a new vector is a linear combination

$$\underline{C_1\vec{a}+C_2\vec{b}}=\begin{bmatrix} c_1a_1 & c_2b_1 \\ \underline{c_1a_2} & \underline{c_2b_2} \end{bmatrix}$$

$\vec{a}, \vec{b}$의 선형 결합 (C_1, C_2는 상수)

Linear combination and linearly independent

❖ Linear combination

- For example, $\vec{a} = \begin{bmatrix} 2 \\ 4 \end{bmatrix}, \vec{b} = \begin{bmatrix} 1 \\ 6 \end{bmatrix}$ $\quad 4\vec{a} + (-3)\vec{b}$

$$4\vec{a} + (-3)\vec{b} = 4\begin{bmatrix} 2 \\ 4 \end{bmatrix} + (-3)\begin{bmatrix} 1 \\ 6 \end{bmatrix} = \begin{bmatrix} 8 \\ 16 \end{bmatrix} + \begin{bmatrix} -3 \\ -18 \end{bmatrix} = \begin{bmatrix} 5 \\ -2 \end{bmatrix}$$

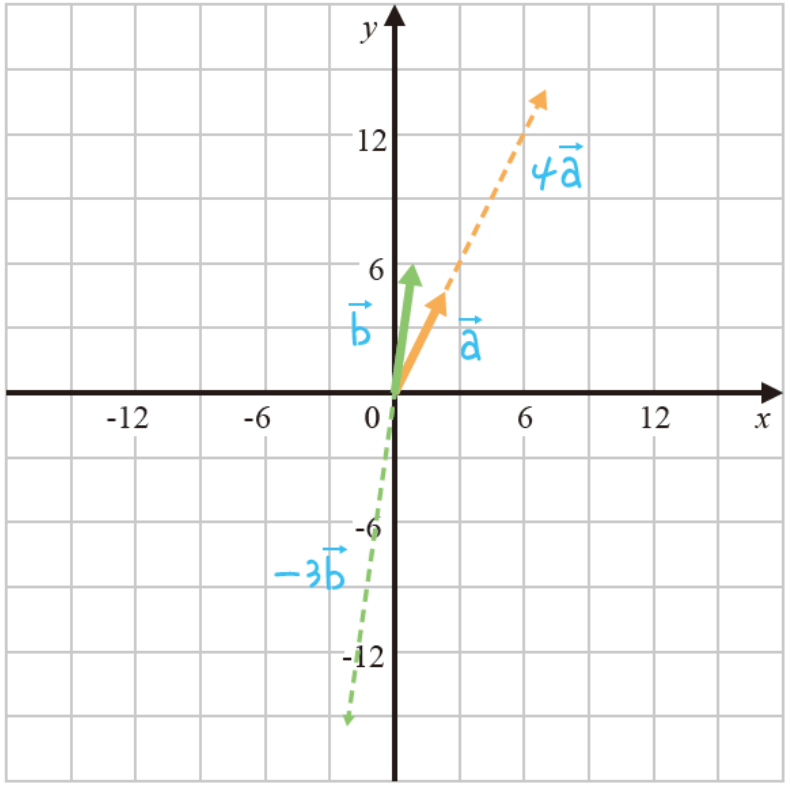
- In other words, a new vector can be created by a constant multiple of the two vectors

$\vec{a}, \vec{b}$

Linear combination and linearly independent

❖ **Linear combination**

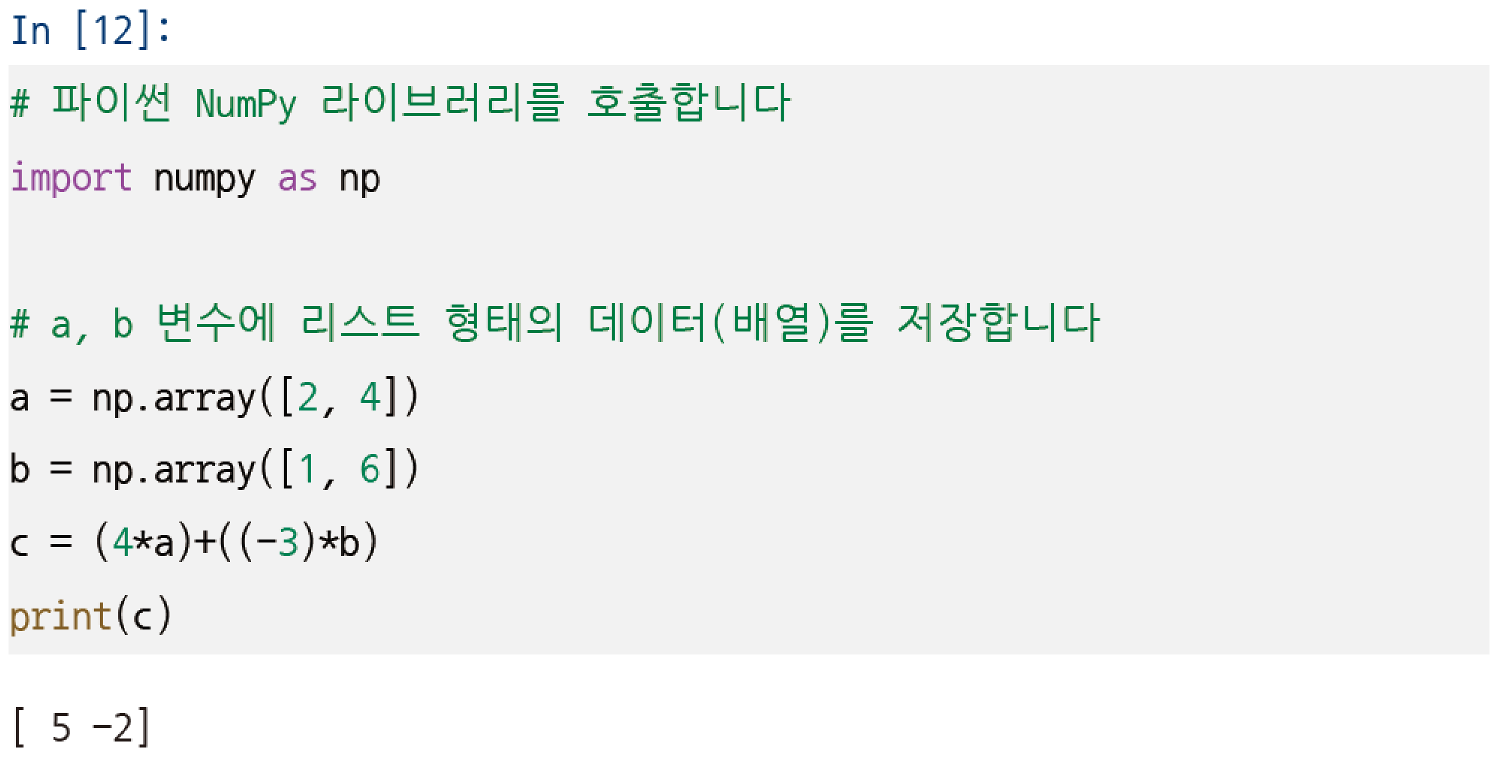
- If you express $\vec{a}=(2,\ 4),\ \vec{b}=(1,\ 6)$ it's like a solid line
- If you express $4\vec{a}=(8,\ 16),\ -3\vec{b}=(-3,\ -18)$ted line



Linear combination and linearly independent

❖ Linear combination

- Since Python does not have a function that can combine linearly, it is necessary to obtain the desired result with the following general operations



In [12]:

```
# 파이썬 NumPy 라이브러리를 호출합니다
import numpy as np

# a, b 변수에 리스트 형태의 데이터(배열)를 저장합니다
a = np.array([2, 4])
b = np.array([1, 6])
c = (4*a)+((-3)*b)
print(c)
```

```
[ 5 -2]
```

Linear combination and linearly independent

MATHEMATIC

❖ Linearly independent and dependent

Linearly independent

- Unlocking and defining linear independents makes them difficult to understand and complex
- To define linear independence with a formula
- For example, for **A = {a_1, a_2, a_3,⋯, a_n} , c_1a_1, + c_2a_2, + c_3a_3, + …, + c_na_n = 0 is called linearly independent when all c is 0**
- That is, the constant value that satisfies this equation, if $c_1 = c_2 = c_3 = \ldots = c_n = 0$, a_1, a_2, a_3, …, a_n is called **linearly independent**

$$c_1a_1, + c_2a_2, + c_3a_3, + \ \ldots, + c_na_n = 0$$

Linear combination and linearly independent

❖ Linearly independent and dependent

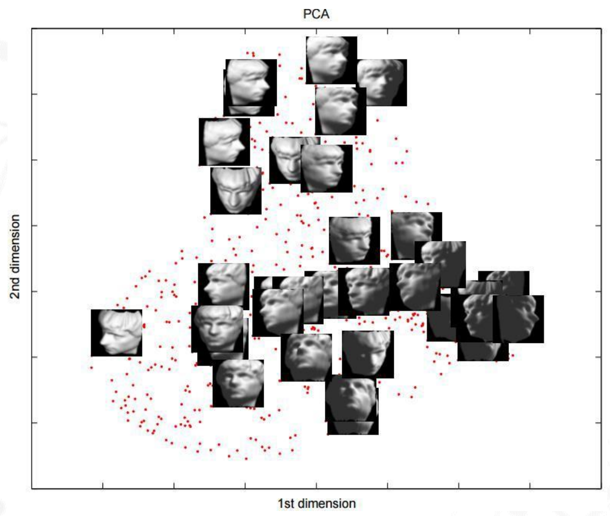
How do you use Linearly independent and linearly dependent in AI?

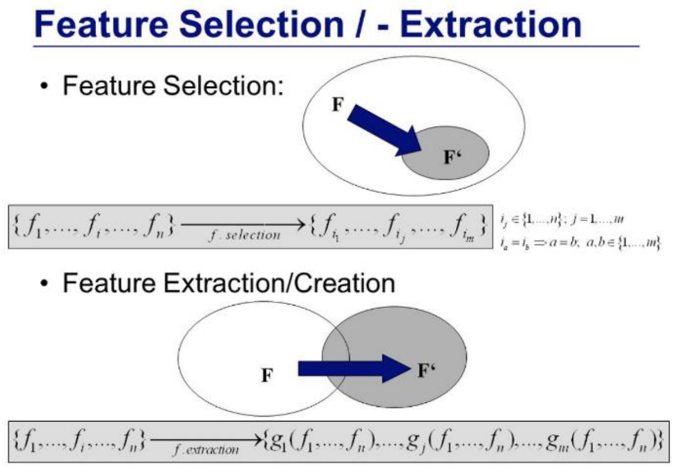
- Overfitting occurs when data set characteristics increase in artificial intelligence
- This phenomenon is called the curse of dimensions, and one of the solutions to this is Principal Component Analysis (PCA)
- PCA uses a feature extraction method, which is a combination of original characteristics to create new characteristics
- This means finding the main axis in the data and projecting all the data onto that axis (as a linear combination of the original properties) to create new properties
- In this case, characteristic extraction works well when the relationship between the characteristics of the data is a linear dependency relationship

Dimension Reduction

❖ Dimension reduction
  - Converting high-dimensional data to low dimensions with minimal loss of information

❖ Purpose
  - 2D or 3D conversion and visualization enables intuitive data analysis
  - Mitigate the issue of curses of dimensionality





Feature Selection / - Extraction

- Feature Selection:

$$\{f_1,\ldots,f_i,\ldots,f_n\} \xrightarrow[f.\,selection]{} \{f_{i_1},\ldots,f_{i_j},\ldots,f_{i_m}\}$$

$i_j \in \{1,\ldots,n\};\ j=1,\ldots,m$

$i_a = i_b \Rightarrow a=b,\ a,b \in \{1,\ldots,m\}$

- Feature Extraction/Creation

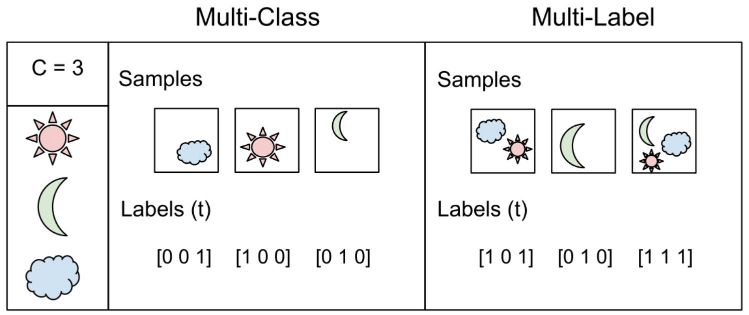
$$\{f_1,\ldots,f_i,\ldots,f_n\} \xrightarrow[f.\,extraction]{} \{g_1(f_1,\ldots,f_n),\ldots,g_j(f_1,\ldots,f_n),\ldots,g_m(f_1,\ldots,f_n)\}$$

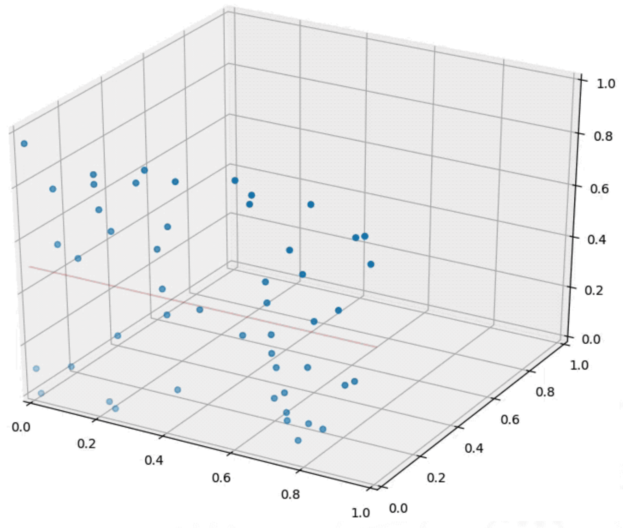
Image: Ghodsi 2006

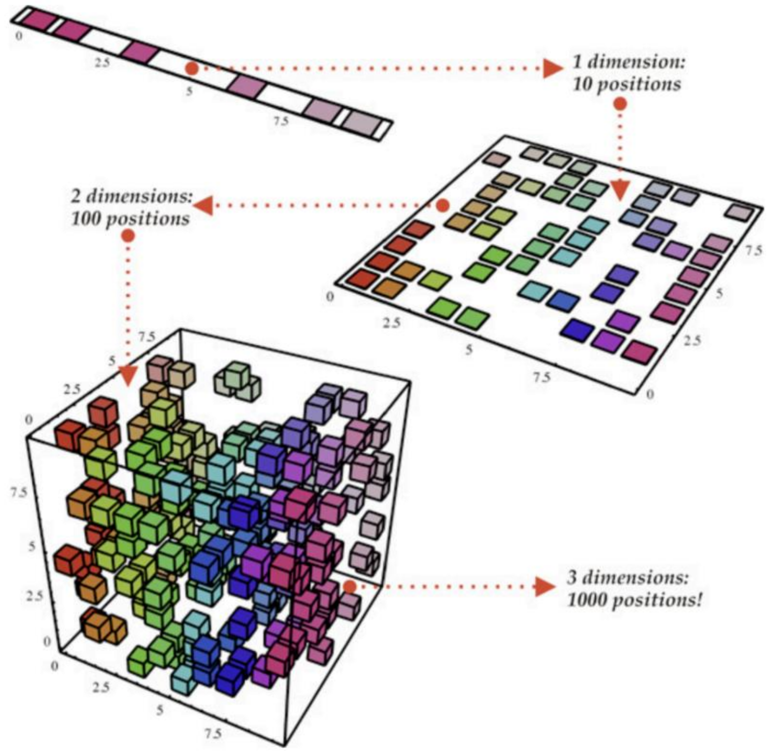
Dimension Reduction

❖ Curse of dimension

- The tendency of distance distribution to be constant as dimension increases
- As dimensions increase, the number of subspaces increases exponentially



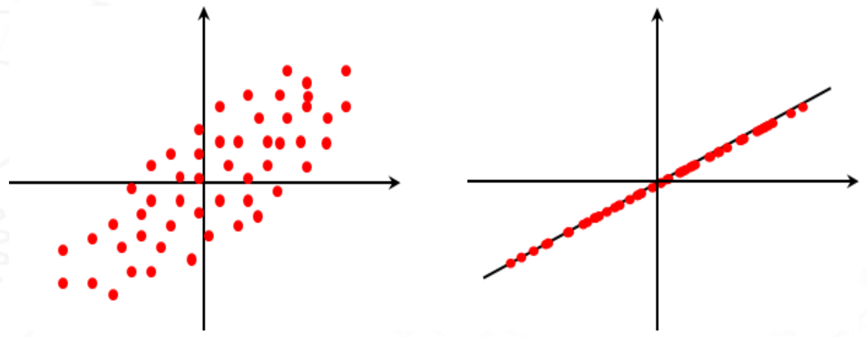


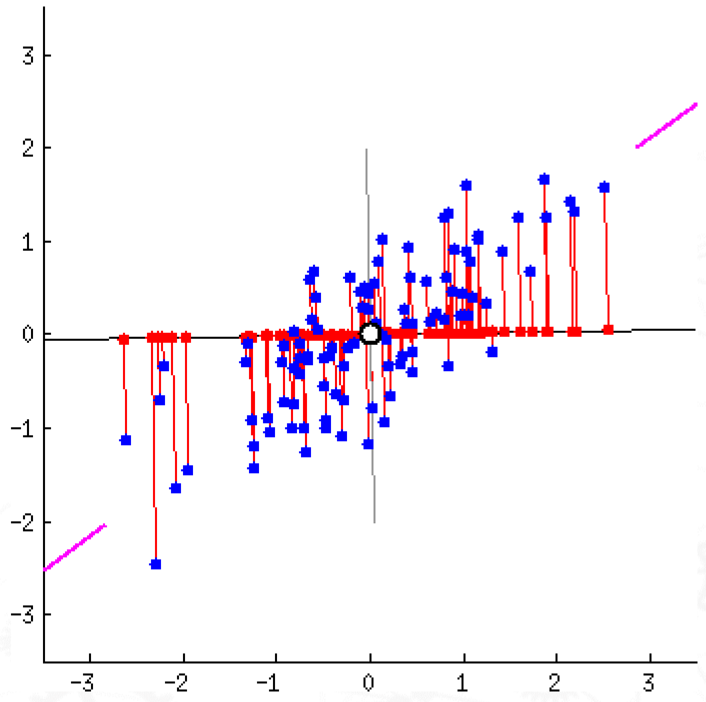


KDD 2014 tutorial, Yoshua Bengio

Dimension Reduction

- Principle Component Analysis, PCA
  - Project data based on several large distributed axes and convert it into low dimensions
  - Choose a few eigenvectors with a large eigenvalue for the covariance matrix of the data as the event axis





https://builtin.com/data-science/step-step-explanation-principal-component-analysis