

# 인공지능 기초

## 인공지능\_ Day05

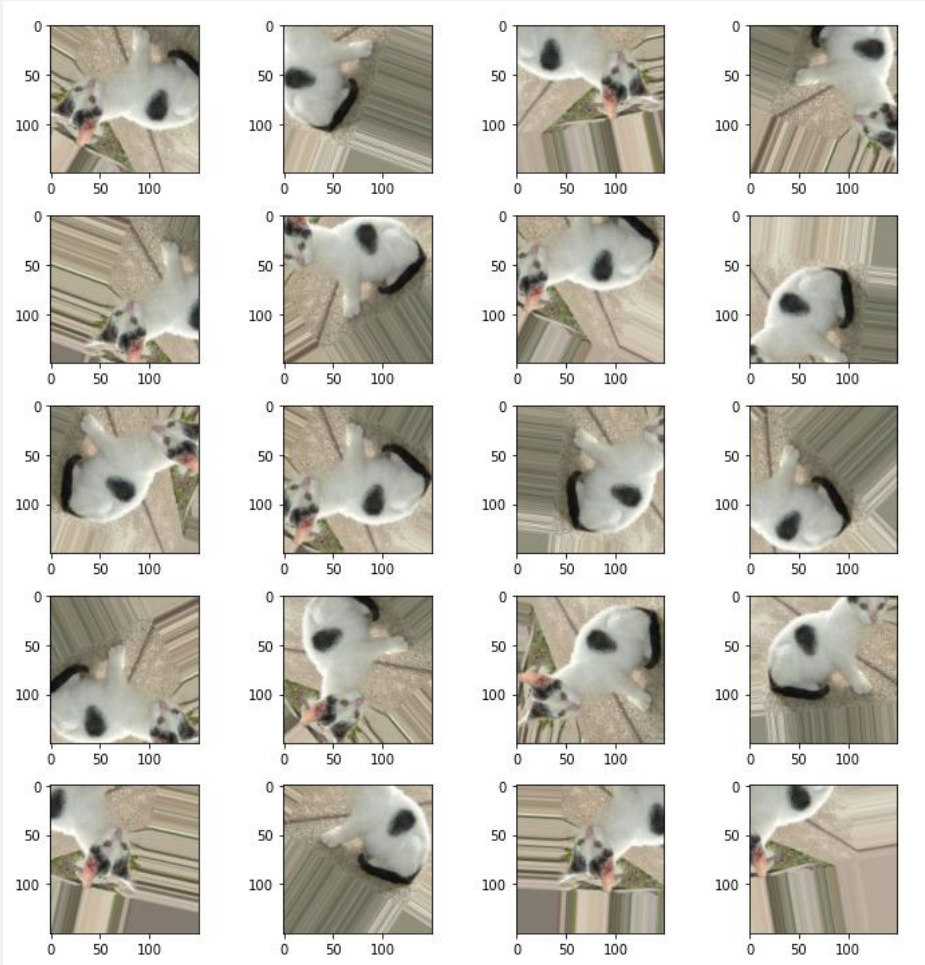
ImageDataGenerator	01
자연어처리 기초	02
문장의 감성분류	03
실습	04

**IDG(ImageDataGenerate)**

# ImageDataGenerator란?

- 이미지를 학습시킬 때 학습데이터의 양이 적을 경우  
학습데이터를 조금씩 변형시켜서 학습데이터의 양을 늘리는 방식
- 과정합을 방지하고 학습 데이터의 다양성을 늘리기 위해 사용
- 비용과 시간을 절약하면서 학습 데이터의 다양성을 늘릴 수 있음
- ImageDataGenerator 의 역할은 이미지 데이터의 증강과 이미지의 전처리
- 공식 문서 <https://keras.io/ko/preprocessing/image/> 참고

# ImageDataGenerator란?



## 공간 레벨 변형

Flip : 상하, 좌우 반전

Rotation : 회전

Shift : 이동

Zoom : 확대, 축소

Shear : 기울이기

## 픽셀 레벨 변형

Bright : 밝기 조정

Channel Shift : RGB 값 변경

## 사용 Library

```
import numpy as np
import matplotlib.pyplot as plt

from keras.preprocessing.image import ImageDataGenerator
```

## 학습 이미지에 적용한 augmentation 인자를 지정


```
train_datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True)
```

## flow\_from\_directory() 함수 사용 - 라벨 정보 포함 이미지를 불러옴

```
train_generator = train_datagen.flow_from_directory(  
    `data/train`, # this is the target directory  
    target_size=(150, 150),  
    batch_size=batch_size,  
    class_mode=`binary`)  
  
validation_generator = validation_datagen.flow_from_directory(  
    `data/validation`,  
    target_size=(150, 150),  
    batch_size=batch_size,  
    class_mode=`binary`)
```

# 자연어처리기초



- 텍스트를 컴퓨터가 이해할 수 있도록 숫자로 변환
- 단어를 표현하는 방법에 따라서 자연어 처리의 성능이 크게 달라짐
- 워드 임베딩은 각 단어를 인공 신경망 학습을 통해 벡터(Vector)화하는 방법
- 케라스에서 제공하는 Embedding()  단어를 랜덤한 값을 가지는 벡터로 변환한 뒤에, 인공 신경망의 가중치를 학습

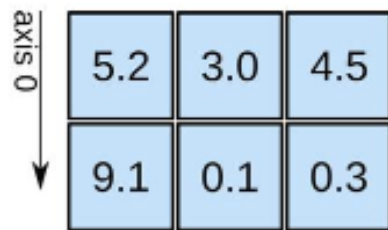
- 인공지능 분야에서 벡터는 대개 고차원의 숫자 배열(array)을 의미함
- 이미지 데이터의 벡터는 각 픽셀(pixel)의 색상 값을 숫자로 표현하고, 이러한 숫자들을 배열 형태로 나열한 것  
(예를 들어, 28 x 28 픽셀의 흑백 이미지는 784차원의 벡터)
- 텍스트 데이터의 벡터는 각 단어(word)를 고유한 정수 인덱스(index)로 매핑하여, 이를 순서대로 배열한 것 → 단어 간의 유사도 계산, 문서 간의 유사도 비교하는 등 다양하게 활용

1D array



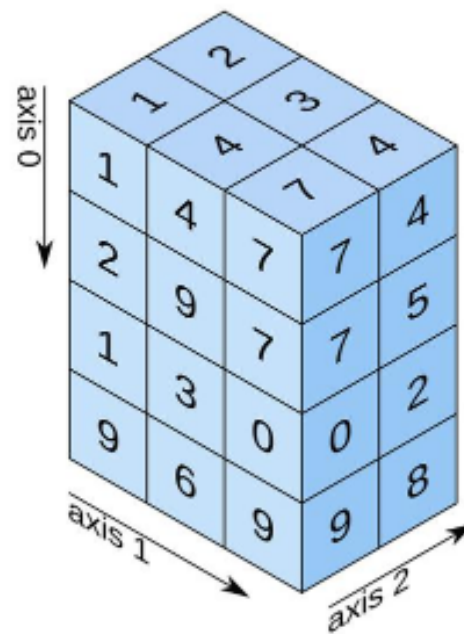
shape: (4,)

2D array



shape: (2, 3)

3D array

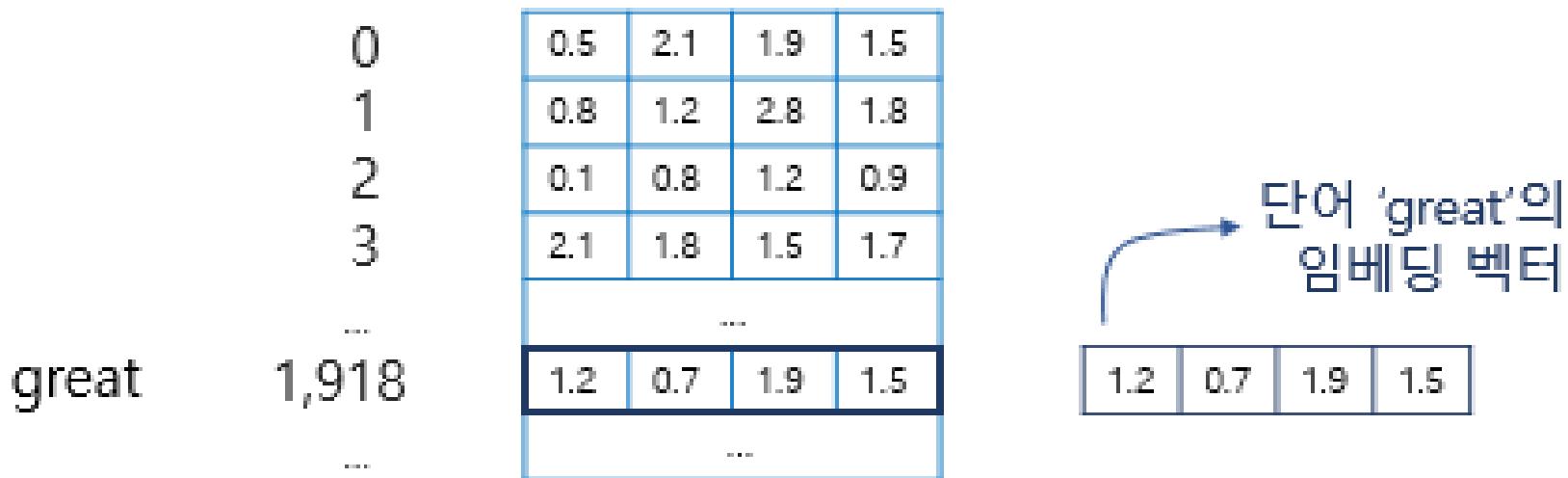


shape: (4, 3, 2)

케라스의 Embedding() - 훈련 데이터의 단어들에 대해 워드 임베딩을 수행

단어 → 단어에 부여된 고유한 정수값 → 임베딩 층 통과 → 밀집 벡터

**Word → Integer → lookup Table → Embedding vector**



훈련 과정에서 학습된다.

## 문장의 긍, 부정을 판단하는 감성 분류 모델

```
import numpy as np
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

sentences = ['nice great best amazing', 'stop lies', 'pitiful nerd',
             'excellent work', 'supreme quality', 'bad', 'highly respectable']
y_train = [1, 0, 0, 1, 1, 0, 1] → 긍정적인 문장은 1, 부정인 문장은 0
```

## 문장의 긍, 부정을 판단하는 감성 분류 모델

### 1. 케라스의 토크나이저를 사용하여 단어 집합을 만들고 크기 확인

```
tokenizer = Tokenizer()  
tokenizer.fit_on_texts(sentences)  
vocab_size = len(tokenizer.word_index)
```

### 2. 각 문장에 대해서 정수 인코딩을 수행

```
X_encoded = tokenizer.texts_to_sequences(sentences)
```

## 문장의 긍, 부정을 판단하는 감성 분류 모델

### 3. 가장 길이가 긴 문장의 길이 계산

```
max_len = max(len(l) for l in X_encoded)
```

### 4. 최대 길이로 모든 샘플에 대해서 패딩 진행

```
X_train = pad_sequences(X_encoded, maxlen=max_len, padding='post')  
y_train = np.array(y_train)  
print('패딩 결과 :')  
print(X_train)
```

## 문장의 긍, 부정을 판단하는 감성 분류 모델

### 5. 전형적인 이진 분류 모델을 설계

```
from tensorflow.keras.models import Sequential
```

```
from tensorflow.keras.layers import Dense, Embedding, Flatten
```

```
embedding_dim = 4
```

```
model = Sequential()
```

```
model.add(Embedding(vocab_size, embedding_dim, input_length=max_len))
```

```
model.add(Flatten())
```

```
model.add(Dense(1, activation='sigmoid'))
```

→ output에 1개의 뉴런을 배치  
활성화 함수는 시그모이드 함수



## 문장의 긍, 부정을 판단하는 감성 분류 모델

### 5. 전형적인 이진 분류 모델을 설계

```
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['acc'])
```

```
model.fit(X_train, y_train, epochs=100, verbose=2)
```

→ 손실 함수로 binary\_crossentropy 사용

실습

1. Tensorflow Server 설정
2. ImageDataGenerator - Cat or Dogs
3. ImageDataGenerator - Horse or Human
4. ImageDataGenerator - rps
5. ImageDataGenerator - brain
6. Word embedding

## Day05. 인공지능 Study

1. 인공지능 개념 정리 - 머신러닝, 딥러닝
2. 퍼셉트론 (Perceptron)
3. 다층 퍼셉트론 (Multi-Layer Perceptron: MLP)
4. 옵티마이저 (Optimizer)
5. 학습률 (learning rate)
6. 경사하강법 (Gradient Descent)
7. 손실함수 (Loss Function)
8. 활성화 함수 (Activation Function) - Sigmoid, ReLU, Softmax

- 9. 회귀분석
- 10. 결정계수 R2 score
- 11. 분류분석
- 12. 원 핫 인코딩 (One Hot Encoding)
- 13. 난수값 (random\_state)
- 14. 정확도 accuracy score
- 15. 과적합 (overfitting)
- 16. 합성곱신경망(CNN)
- 17. 이미지증강(ImageDataGenerator)

## 18. 자연어처리(Word Embedding)

수고하셨습니다.