



딥러닝 실습

인공지능 RPA - 13주차

학습 내용

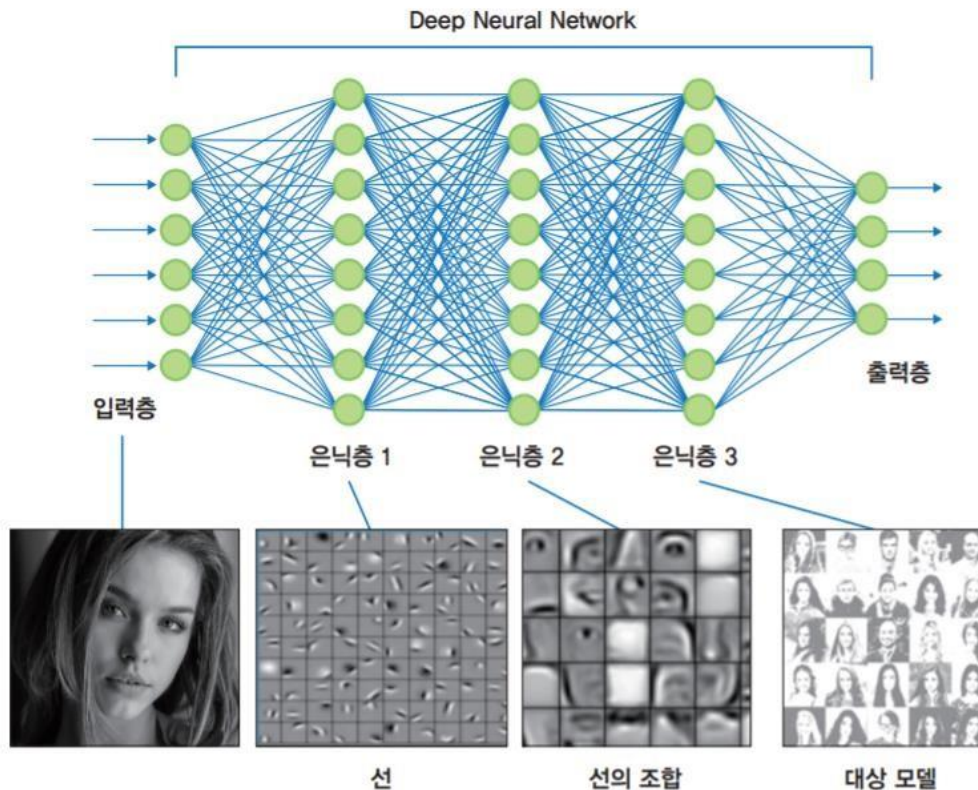
1. 딥러닝 실습
2. FastAPI - DB 연결 처리



인공지능 실습

_딥러닝

- 딥러닝(Deep Learning) : 머신러닝 기법 중 하나인 인공신경망(Artificial Neural Networks)기법의 은닉층(Hidden Layer)을 깊게 쌓은 구조를 이용해 학습하는 기법
- 딥러닝의 장점 : 데이터의 특징을 단계별로 추상화를 높여 가면서 학습할 수 있음
- 얇은 은닉층은 점,선,면과 같은 추상화 단계가 낮은 특징을 학습하고, 깊은 은닉층은 얼굴의 눈,코,입 등 추상화 단계가 높은 특징을 학습한다. (세부 특징 → 상위 특징(세부 특징의 조합))
- 딥러닝을 사용할 경우 사람처럼(추상화 단계가 높은 특징을 사용해서 판단) 고차원적 인지활동을 수행할 수 있음



기존 신경망과 비교해서 은닉층과 출력층이 2개 이상

_단순 신경망과 딥러닝의 차이점



강아지



강아지



강아지

_단순 신경망과 딥러닝의 차이점



눈! 코! 귀! 다있네요!
강아지 확률 97%!



눈! 코! 귀! 다있네요!
강아지 확률 98%!

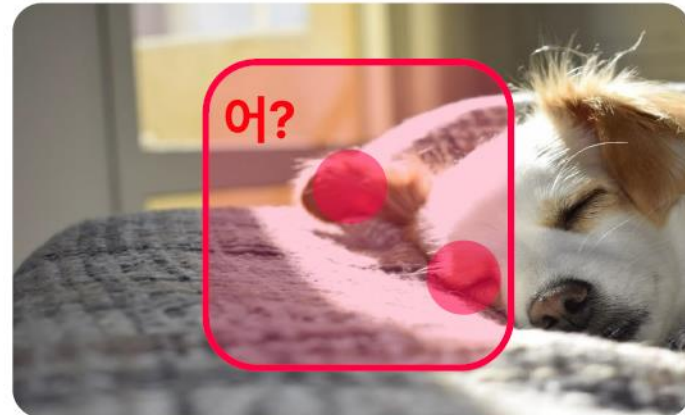


눈! 코! 귀! 다있네요!
강아지 확률 96.5%!

_단순 신경망과 딥러닝의 차이점



강아지-45%



뽀족한 귀와 눈한쪽이 있습니다!
강아지 확률 32%!

_TensorFlow 설치

1. Python 3.9 설치

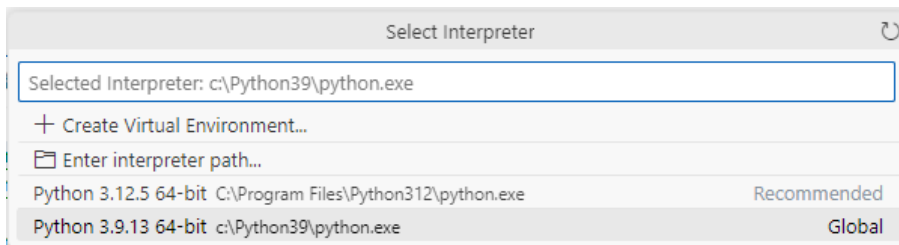
<https://www.python.org/downloads/release/python-3913>

Files

Version	Operating System	Description	MD5 Sum	File Size	PGP
Gzipped source tarball	Source release		eafda83543bad127cdef4d288fdab87	25.1 MB	SIG
XZ compressed source tarball	Source release		5e2411217b0060828d5f923eb422a3b8	18.8 MB	SIG
macOS 64-bit Intel-only installer	macOS	for macOS 10.9 and later, deprecated	671848930809decf27f586ddf98c6e9b	29.6 MB	SIG
macOS 64-bit universal2 installer	macOS	for macOS 10.9 and later	76b63cf623e32cdf27c5033434bd69ce	37.0 MB	SIG
Windows installer (64-bit)	Windows	Recommended	e7062b85c3624af82079794729618eca	27.9 MB	SIG
Windows installer (32-bit)	Windows		46c35b0a2a4325c275b2ed3187b08ac4	26.8 MB	SIG
Windows help file	Windows		c86feba059b340a1de2a9d2ee7059a6d	8.5 MB	SIG
Windows embeddable package (64-bit)	Windows		57731cf80b1c429a0be7133266d7d7cf	8.2 MB	SIG
Windows embeddable package (32-bit)	Windows		fec0bc06857502a56dd1a6aea6488ef8	7.4 MB	SIG

Customize install → 설치 경로 C:\Python39

2. VS Code에서 실행환경 3.9버전 등록



_TensorFlow 설치

3. TensorFlow 설치

& c:/Python39/python.exe -m pip install tensorflow-cpu==2.10.0 numpy==1.23.5 scipy

4. 이미지 적용을 위한 라이브러리 설치

& c:/Python39/python.exe -m pip install Pillow



_학습된 모델로 부터 X-ray 인식 (폐렴/정상 구분)

1. 라이브러리 불러오기

```
import tensorflow as tf
from tensorflow.keras.preprocessing import image
import numpy as np
```

2. 모델 불러 오기

```
model = tf.keras.models.load_model('xray_classification_model.h5')
print(model.summary(), end='\n\n')
```

3. 이미지 로딩 및 전처리

```
image_path = r'person1_virus_11.jpeg' # 분류할 이미지 파일 경로
```

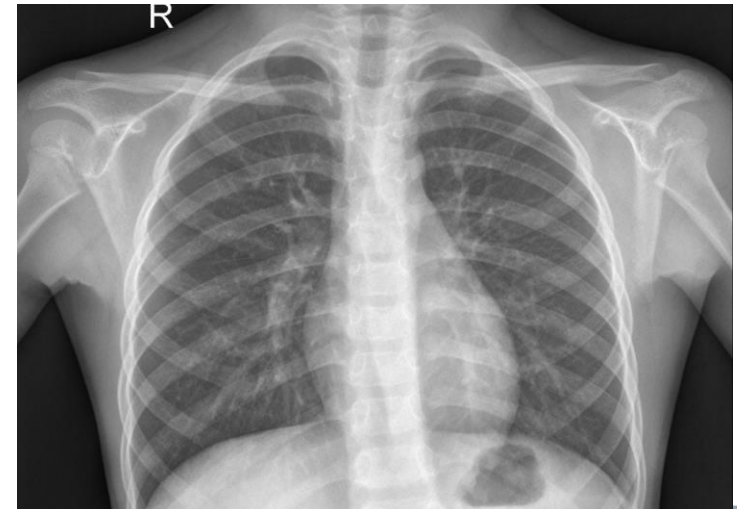
```
img = image.load_img(image_path, target_size=(150, 150))
img_array = image.img_to_array(img)
img_array = np.expand_dims(img_array, axis=0)
img_array /= 255.0 # 이미지를 0-1 범위로 정규화
```

4. 분류 수행

```
predictions = model.predict(img_array)
```

5. 결과 출력

```
print(predictions, end='\n\n')
if predictions[0][0] > 0.5:
    print("이 이미지는 PNEUMONIA에 속합니다.")
else:
    print("이 이미지는 NORMAL에 속합니다.")
```



_학습 모델

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 148, 148, 32)	896
activation (Activation)	(None, 148, 148, 32)	0
max_pooling2d (MaxPooling2D)	(None, 74, 74, 32)	0
conv2d_1 (Conv2D)	(None, 72, 72, 32)	9248
activation_1 (Activation)	(None, 72, 72, 32)	0
max_pooling2d_1 (MaxPooling2D)	(None, 36, 36, 32)	0
conv2d_2 (Conv2D)	(None, 34, 34, 64)	18496
activation_2 (Activation)	(None, 34, 34, 64)	0
max_pooling2d_2 (MaxPooling2D)	(None, 17, 17, 64)	0
flatten (Flatten)	(None, 18496)	0
dense (Dense)	(None, 64)	1183808
activation_3 (Activation)	(None, 64)	0
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 1)	65
activation_4 (Activation)	(None, 1)	0

=====

Total params: 1212513 (4.63 MB)
Trainable params: 1212513 (4.63 MB)
Non-trainable params: 0 (0.00 Byte)

'xray_classification_model.h5

```
model = Sequential()
model.add(Conv2D(32, (3, 3), input_shape=(150, 150, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(32, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(64))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(1))
model.add(Activation('sigmoid'))
```



RPA 실습

_FastAPI

Fast API 데이터베이스 연결

- 클라이언트 코딩(Front)
- 서버 코딩 (Back)

1. 클라이언트 코딩


: static 폴더안에 html 파일 생성

```
<body>  
  <form action="/login" method="post">  
    <span>ID</span>  
    <input type="text" name="userid"><br>  
    <span>PW</span>  
    <input type="text" name="userpassword"><br>  
    <input type="submit" value="로그인">  
  </form>  
</body>
```

ID	<input type="text" value="user1"/>
PW	<input type="text" value="user123"/>
<input type="button" value="로그인"/>	



Sqlite 데이터베이스 보기




SQLite Viewer

v0.8.3

Florian Klampfer | 1,366,954 | ★★★★★ (53)

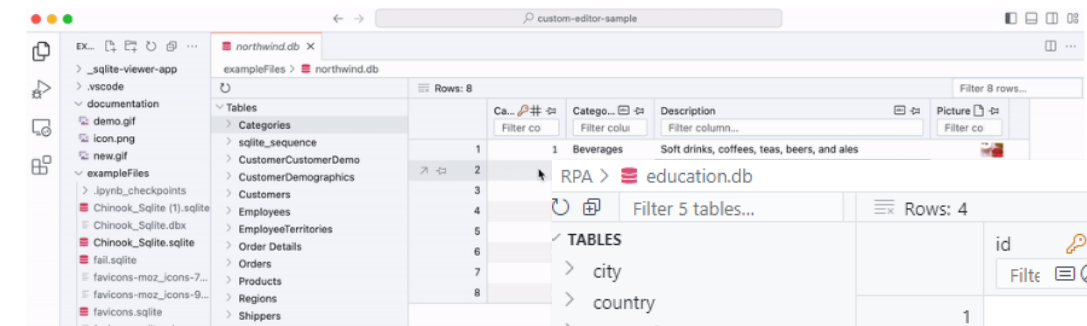
SQLite Viewer for VS Code

[Disable](#) [Uninstall](#) ☒ Auto Update 

[DETAILS](#) [FEATURES](#) [CHANGELOG](#)

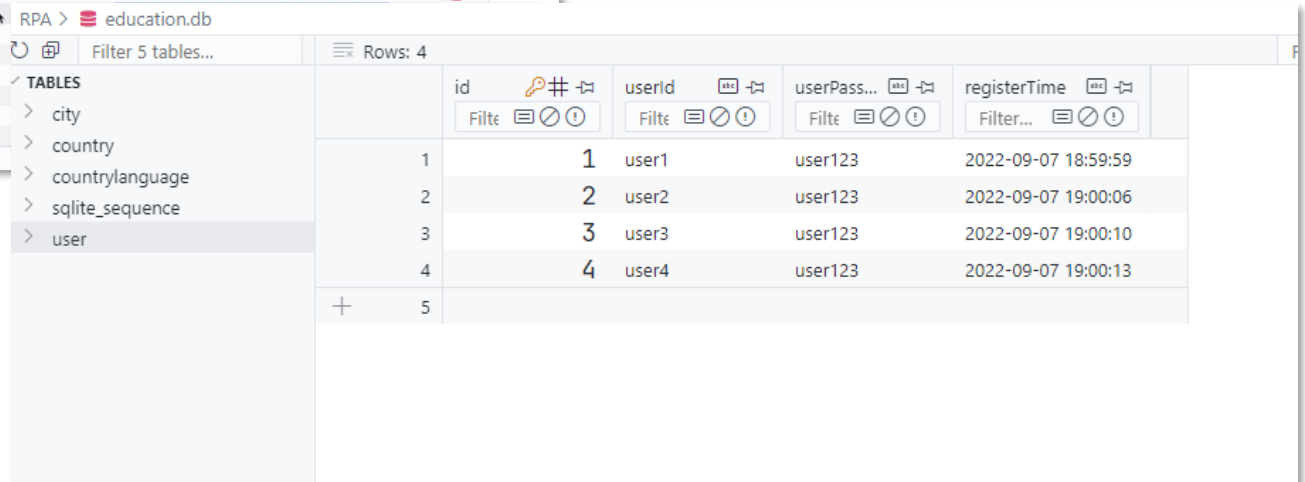
SQLite Viewer for VSCode

A quick and easy SQLite viewer for VSCode, inspired by DB Browser for SQLite and Airtable.



The screenshot shows the SQLite Viewer interface within VS Code. The left sidebar displays a file explorer with various database files. The main panel shows the 'northwind.db' database with a list of tables. The 'Beverages' table is selected, and its data is displayed in a table view.

id	Category	Description	Picture
1	Beverages	Soft drinks, coffees, teas, beers, and ales	



The screenshot shows the SQLite Viewer interface within VS Code. The left sidebar displays a file explorer with various database files. The main panel shows the 'education.db' database with a list of tables. The 'user' table is selected, and its data is displayed in a table view.

id	userid	userPass...	registerTime
1	1	user1	2022-09-07 18:59:59
2	2	user2	2022-09-07 19:00:06
3	3	user3	2022-09-07 19:00:10
4	4	user4	2022-09-07 19:00:13



_FastAPI

2. 서버 코딩 : Form 요청에 대응하는 API

```
@app.post("/login")
def login_form(userid: str = Form(...), userpassword: str = Form(...)):
    result = loginDB (userid, userpassword)

    if result == True :
        return {"msg": f"{userid}님 반갑습니다."}
    else :
        return {"msg": f"로그인에 실패했습니다."}
```



FastAPI

```
def loginDB(userId, userPassword):
```

```
    import sqlite3
```

```
    conn = sqlite3.connect('education.db')
```

```
    cursor = conn.cursor()
```

```
    query = 'SELECT * FROM user WHERE userId = ? AND userPassword = ?'
```

```
    cursor.execute(query, (userId, userPassword))
```

```
    result = cursor.fetchone()
```

```
    conn.close()
```

```
    if result:
```

```
        print("Login successful!")
```

```
        return True
```

```
    else:
```

```
        print("Login failed. Invalid username or password.")
```

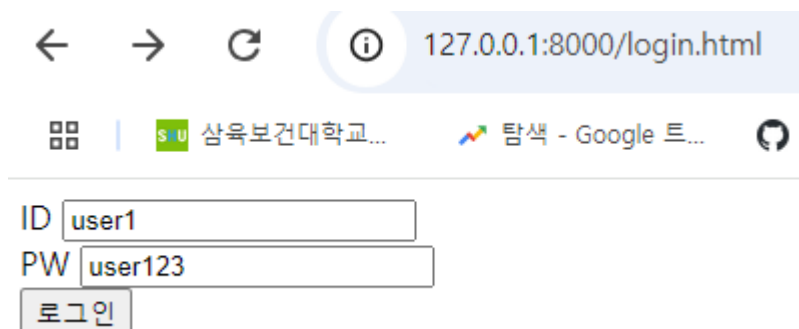
```
        return False
```



_FastAPI

3. 실행 확인

http://127.0.0.1:8000/login.html

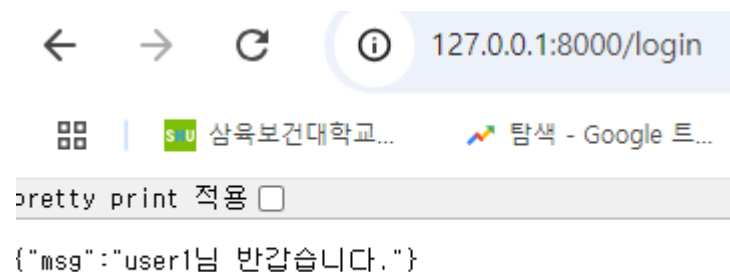


← → ↻ ⓘ 127.0.0.1:8000/login.html

☰ | S+U 삼육보건대학교... 🔍 탐색 - Google 트...

ID

PW



← → ↻ ⓘ 127.0.0.1:8000/login

☰ | S+U 삼육보건대학교... 🔍 탐색 - Google 트...

pretty print 적용 ☐

{"msg": "user1님 반갑습니다."}

