Portland State
UNIVERSITY

Python을 이용한
고강도강 겹치기 레이저 용접부의 모델링

- SNN (shallow), DNN (deep) -

Anaconda Jupyter notebook 사용

안 내

- 본 자료는 아래의 사람들이 만들었습니다.
  유현정 (Portland State University)
  이기동 (Portland State University)
  김철희 (한국생산기술연구원, Portland State University)

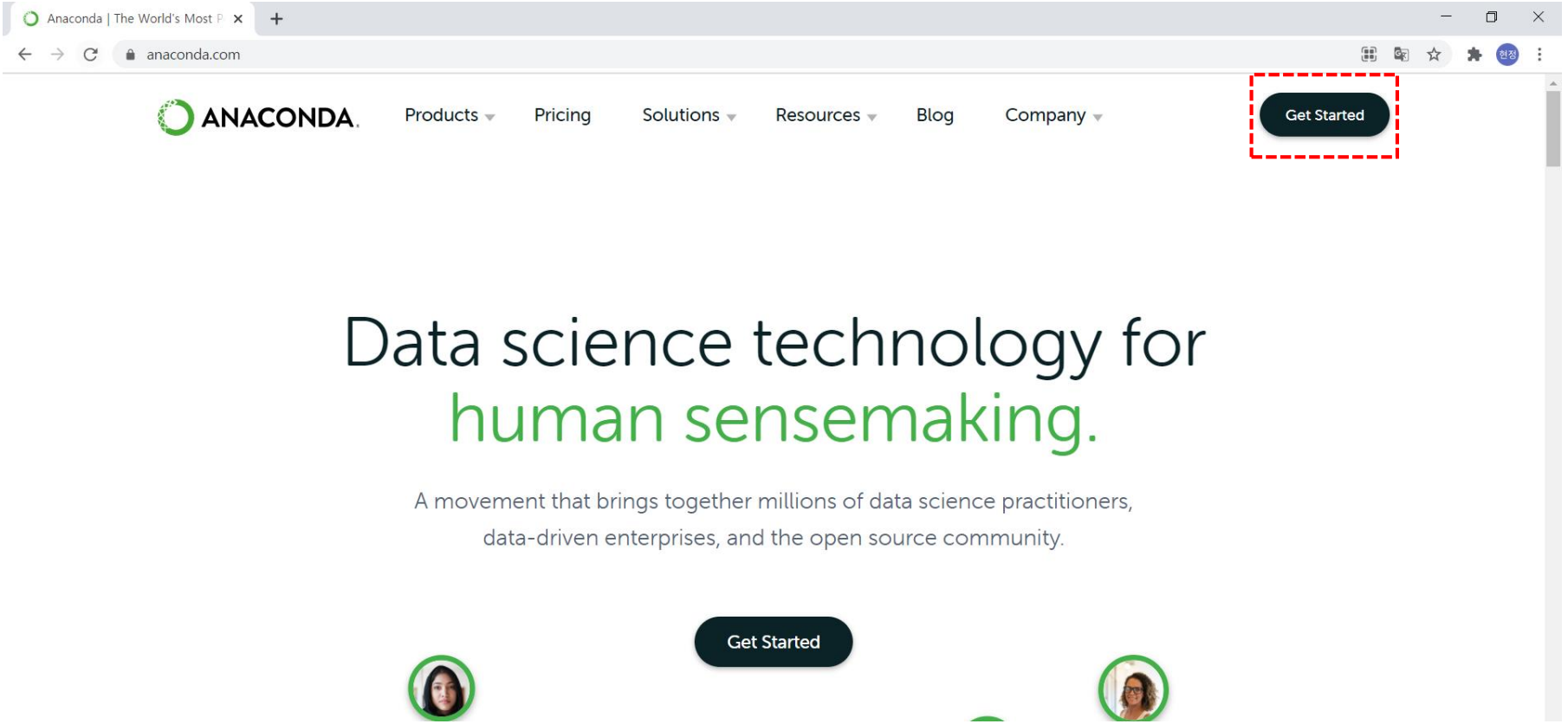
- 예제 파일은 아래에서 받을 수 있습니다.
  https://deepjoining.github.io/

- 문의사항 및 의견: deepjoining@gmail.com

- 자료는 한국생산기술연구원 용접접합그룹 신입대학원생 교육자료입니다.
  일체의 다른 용도 사용을 금지합니다.

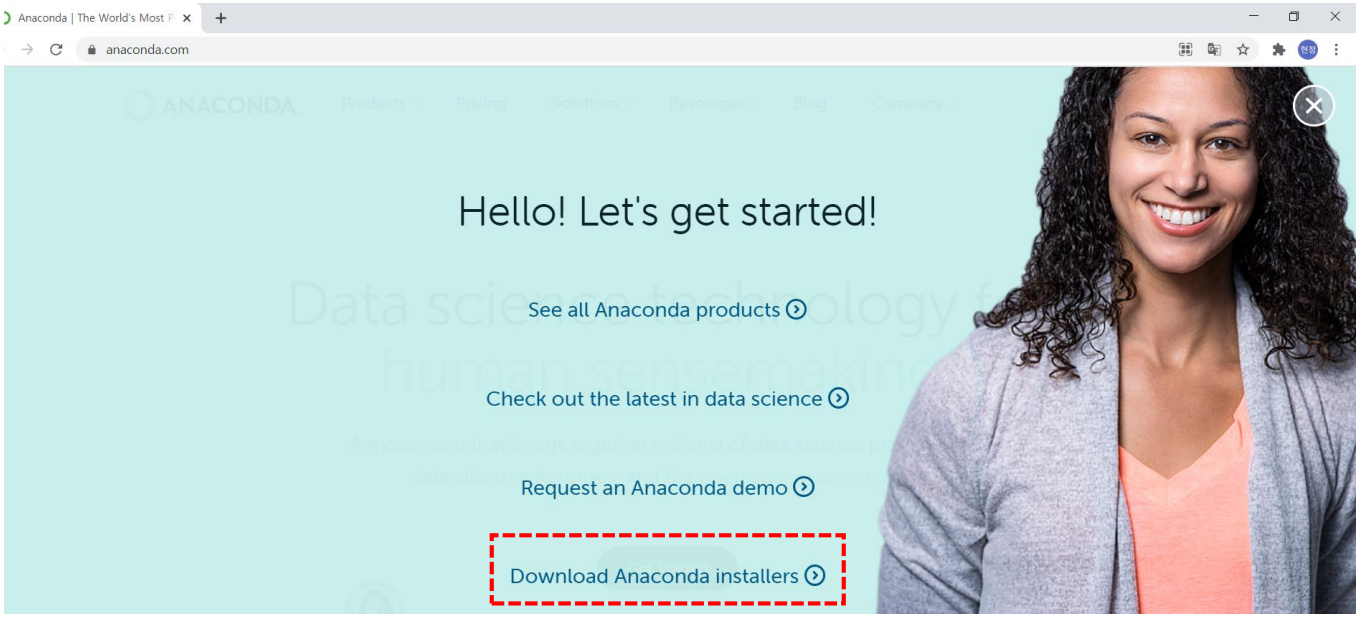
0. Python 설치 방법

- 아나콘다 파일 설치 방법
  - https://www.anaconda.com/ 에 들어간 후 Get started 클릭



0. Python 설치 방법

- Download Anaconda installers 클릭



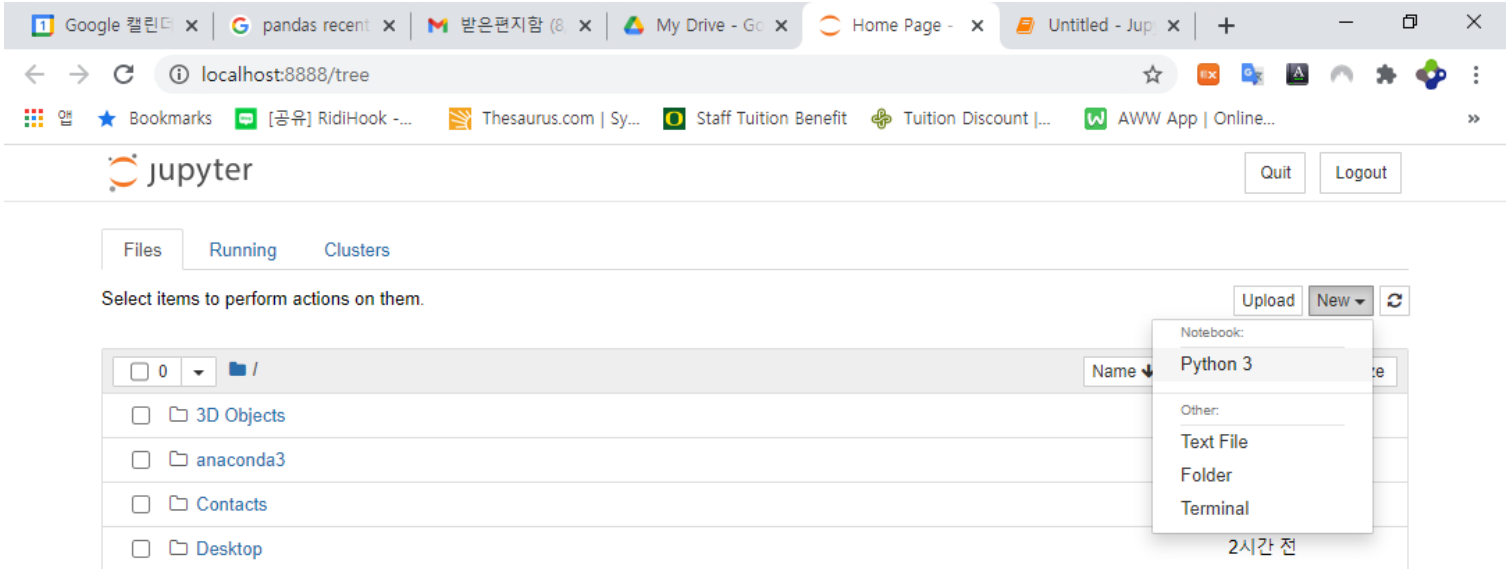
- 컴퓨터 환경에 맞는 파일 설치

0. Python 설치 방법

- PC에서 실행하기
  - Anaconda Prompt 프로그램 실행 후 아래를 설치
    - ✓ pip install tensorflow
    - ✓ pip install keras
    - ✓ pip install pandas==1.2.1 (01/20/21 update)
    - ✓ pip install -U scikit-learn
    - ✓ pip install numpy --upgrade
  - 200인 이상 기업은 유료이므로 생기원 설치 시 유의할 것

1. 딥러닝을 하기 위한 준비

- Jupyter Notebook 실행
  - Python 3 노트북을 하나 열어서 실습한다



  - Shift+enter로 한 줄씩 실행이 가능하다

1. 딥러닝을 하기 위한 준비(https://www.w3schools.com/python/default.asp)

- C:\temp 폴더를 만들고 csv 파일 2개를 복사해서 넣는다.
- Regression이나 classification의 Keras code를 붙여서 넣고 shift+enter를 누른다.
- Python에서 List의 slicing

List = [0, 1, 2, 3, 4, 5]

0	1	2	3	4	5

List[0] = 0

List[1] = 1

List[2] = 2

List[3] = 3

List[4] = 4

List[5] = 5

List[0:] = [0,1,2,3,4,5]

List[:] = [0,1,2,3,4,5]

List[2:4] = [2, 3]

List[1:3] = [1, 2]

List[:4] = [0, 1, 2, 3] 4를 포함하지 X

List = [0, 1, 2, 3, 4, 5]

Forward Direction → 0 1 2 3 4 5

0	1	2	3	4	5

-6 -5 -4 -3 -2 -1 ← Backward Direction

```
list = [1,2,3,4,5]
print(list[-1])
print(list[-3:])
print(list[:-1])
print(list[-3:-1])
```

마지막 원소 포함 X

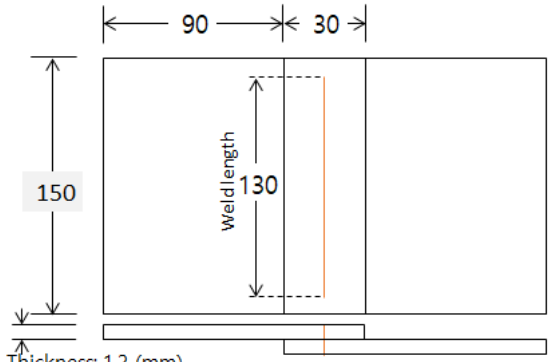
Output:

```
5
[3, 4, 5]
[1, 2, 3, 4]
[3, 4]
```

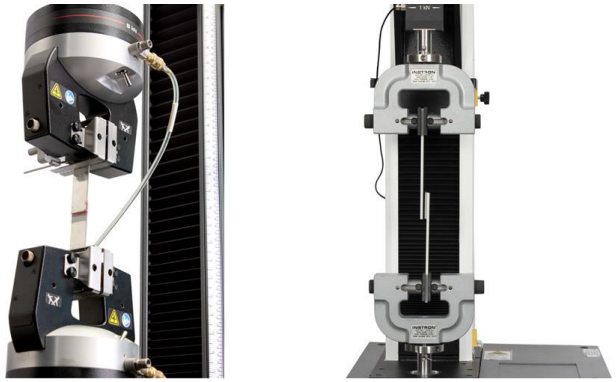
2. 풀어야 할 문제

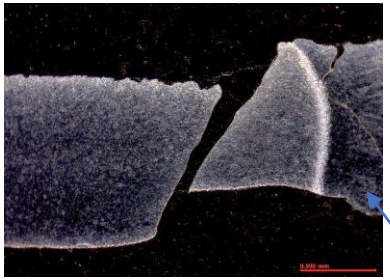
- 소재: 인장강도 590~1500 MPa급 자동차용 강판
  (cf. 연강의 경우 인장강도 270~300 MPa)
- 용접방법: 레이저 겹치기 용접
- 용접부 시험방법: 인장-전단 강도 평가
- 품질판단 기준: 파단의 위치

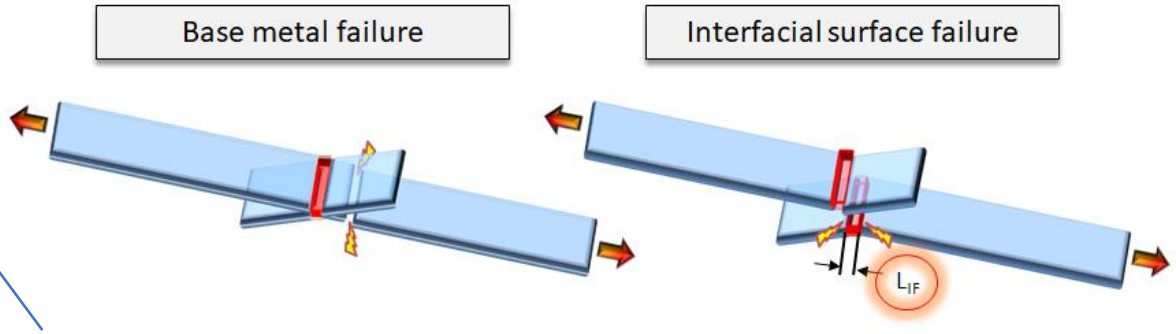
- 모델링할 문제
  * 다양한 소재 조합 및 다양한 레이저 용접조건하에서
  (1) 용접 후 용접 비드폭은 얼마인가? (회귀)
  (2) 인장-전단 시험에서 강도는? (회귀)
  (3) 인장-전단 시험에서 파단위치는? (분류)

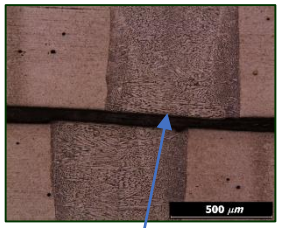


용접 실험 모식도


인장-전단 시험


HPF2.0G
용접부





용접 계면파단

2. 머신 러닝 모델 구축에 사용된 Input, Output parameter

- Input parameter

No.	1	2	3	4
Input parameter	Strength of the upper sheet	Strength of the lower sheet	Welding speed	Focal position

- Output parameter

	Regression model		Classification model
Output parameter	5	6	7
	Bead width at the faying surface	Fracture load	Fracture location

- 엑셀 파일에 정리된 데이터

#1. Strength of upper	#2. Strength of lower	#3. Welding speed	#4. Focal position	#5. Bead width at fayi	#6. Fracture load	#7. Fracture location (
590	590	70	0	0.82	15659.46667	0
590	590	60	-5	0.87	16660.73333	0
590	590	48	-10	1.02	18593.63333	0
590	590	37	-15	1.33	18619.1	2
590	590	26	-20	1.99	18859.83333	2
590	590	20	-25	2.02	18765.03333	2
590	780	70	0	0.76	14681.73333	0
590	780	60	-5	0.81	15620.36667	0

3. Python을 이용한 분류문제 코드: PC 환경

```
# Library, function 불러오기

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

import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation
from keras.optimizers import Adam
from keras import backend as K
from keras.utils import to_categorical
```

3. Python을 이용한 분류문제 코드: PC 환경

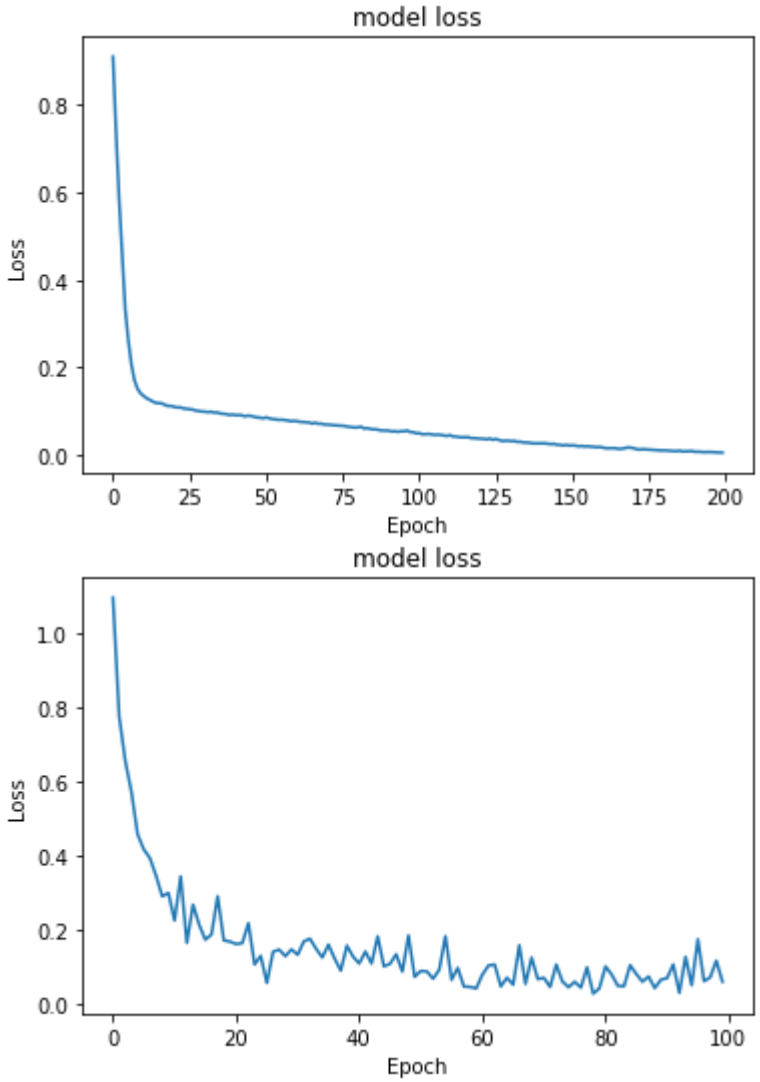
```
# 데이터 불러오기 및 전처리

Xy = np.loadtxt( ' C:/temp/data_laser_2.csv ' , delimiter= ' , ' , dtype=np.float32)

# 가장 마지막 column이 분류, 나머지는 Input 변수
X_data = xy[:, 0:-1]

# 변수x=변수y의 경우 값이 복사되는 것이 아니라 변수가 참조
X_org = copy.deepcopy(x_data)

# Normalizing data
x_data -= x_data.mean(axis=0)
x_data /= x_data.std(axis=0)

# 가장 마지막 column만 추출
Y_data = xy[:, [-1]]

# One hot encode
y_data = to_categorical(y_data)
```

3. Python을 이용한 분류문제 코드: PC 환경

```
# Sequential로 네트워크를 쌓는다
model = Sequential()

model.add(Dense(64, activation='relu', input_dim=4))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(3, activation='softmax'))
model.summary()

# 최적화 함수 및 손실함수 정의
Adam = Adam(lr=0.01, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0, amsgrad=False)
model.compile(loss='categorical_crossentropy', optimizer=adam, metrics=['accuracy'])

# 100 epochs 동안 최적화 실시
history=model.fit(x_data, y_data, epochs=100)
```

3. Python을 이용한 분류문제 코드: PC 환경

```
# 학습 결과 확인


# check results.
# 화면에 실제값과 예측값 비교 가능
For I in range(len(x_data)):
    print(x_org[I,:], y_data[I,:], model.predict(x_data)[i])


# Graph 그리기 (교재 3-7)
plt.plot(history.history['Loss'])
plt.title('Training loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.show()
```



감 사 합 니 다

다음 강의 내용 : 파이썬을 이용한 SNN, DNN 모델 분석 방법(2)