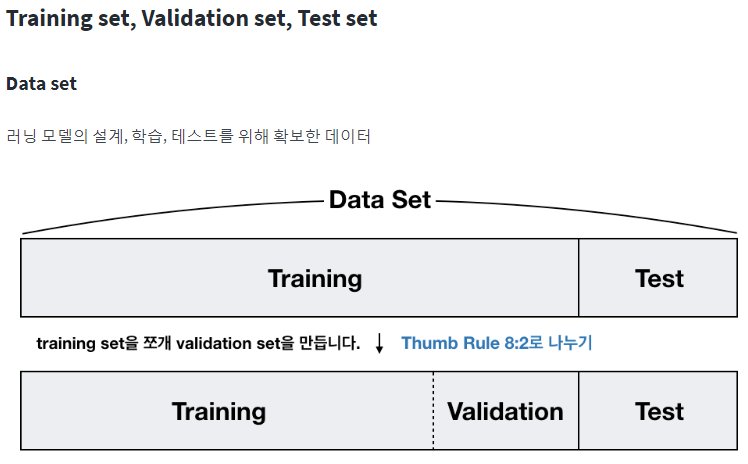
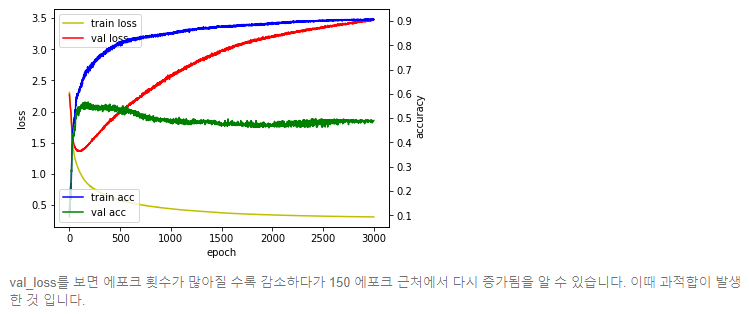
**데이터셋(Data set)**



Training set(훈련셋) : 모델의 학습에 사용되는 데이터.

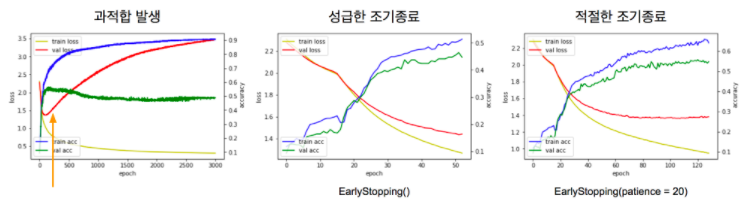
Validation set(검증셋) : 모델 제작 과정 중, 학습된 모델의 성능을 측정하기 위한 데이터.

Test set(시험 셋) : 모델의 최종성능으 평가하기 위한 데이터.

**과적합(Overfitting)**

**학습 조기종료**

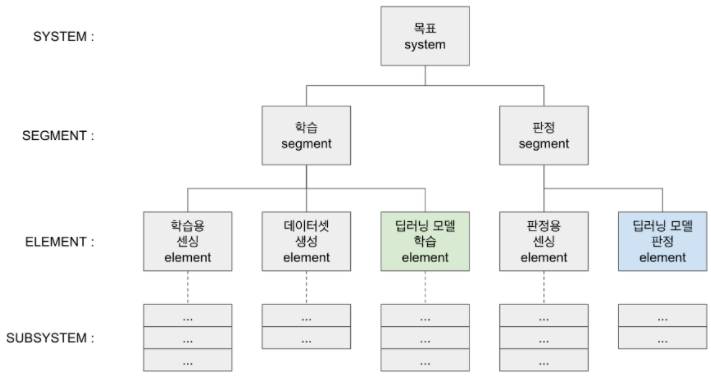
: EarlyStopping함수를 사용



**실무에서의 딥러닝 시스템**

모듈을 분리하기 전에 실무에서의 딥러닝 시스템을 살펴보자.

도메인, 문제마다 다양한 구성이 있겠지만, 기본 딥러닝 시스템 구성은 다음과 같다.



우리가 만들고자 하는 전체 시스템을 목표 시스템이라고 했을 때, 크게 **학습**(segment)과 **판정**(segment)으로 나누어진다.

**학습**은 학습을 위해, 학습데이터를 얻기위한 「학습용센싱 element」, 센싱 데이터에서 학습에 적합한 형태로 전처리를 수행하는 「데이터셋 생성 element」, 그리고 데이터셋으로 딥러닝 모델을 학습시키는 「딥러닝 모델학습 element」으로 나누어진다.

**판정**은 실무환경에서 수집되는 센서인 「판정용 센싱 element」과 학습된 딥러닝 모델을 이용해서 센싱데이터를 판정하는 「딥러닝 모델판정 element」으로 나누어진다. 앞서 본 코드에는 「딥러닝 모델학습 element」와 「딥러닝 모델판정 element」가 모두 포함되어 있다. 이 두가지 element를 분리해보겠다.

**원-핫 인코딩(One-Hot Encoding)**

컴퓨터 또는 기계는 문자보다는 숫자를 더 잘 처리 할 수 있다. 이를 위해 자연어 처리에서는 문자를 숫자로 바꾸는 여러가지 기법들이 있다. **원-핫 인코딩(One-Hot Encoding)**은 그 많은 기법 중에서 단어를 표현하는 가장 기본적인 표현 방법이며, 머신 러닝, 딥 러닝을 하기 위해서는 반드시 배워야 하는 표현 방법이다.

원-핫 인코딩에 대해서 배우기에 앞서 **단어집합(vocabulary)**에 대해서 정의해보도록 하겠다. 어떤 분들은 사전(vocabulary)이라고도 부르지만, 저는 집합이라는 표현이 보다 명확하다고 생각하여 앞으로 단어집합이라고 부르겠다. 단어집합은 앞으로 자연어 처리에서 계속 나오는 개념이기 때문에 여기서 이해하고 가야한다. 단어집합은 서로 다른 단어들의 집합이다. 여기서 혼동이 없도록 서로 다른단어라는 정의에 대해서 좀 더 주목할 필요가 있다.

단어집합(vocabulary)에서는 기본적으로 book과 books와 같이 단어의 변형형태도 다른 단어로 간주한다. 앞으로 단어집합에 있는 단어들을 가지고, 문자를 숫자(더 구체적으로는 벡터)로 바꾸는 원-핫 인코딩을 포함한 여러 방법에 대해서 배우게 된다.

원-핫 인코딩을 위해서 먼저 해야할 일은 단어집합을 만드는 일이다. 텍스트의 모든 단어를 중복을 허용하지 않고 모아놓으면 이를 단어집합이라고 한다. 그리고 이 단어 집합에 고유한 숫자를 부여하는 정수 인코딩을 진행한다. 텍스트에 단어가 총 5,000개가 존재한다면, 단어 집합의 크기는 5,000이다. 5,000개의 단어가 있는 이 단어 집합의 단어들마다 1번부터 5,000번까지 인덱스를 부여한다고 해보자. 가령, book은 150번, dog는 171번, love는 192번, books는 212번과 같이 부여할 수 있다.

이제 각 단어에 고유한 정수 인덱스를 부여하였다고 가정하면, 이 숫자로 바뀐 단어들을 벡터로 다루고 싶다면 어떻게 하면 될까?

**케라스의 Sequential모델 만들기**

**model = Sequential()**

Sequential모델 오브젝트를 model이라는 변수 안에 넣고, 모델 구성을 시작한다.

**model.add(Dense(5, input\_dim=1, activation='relu'))**

「input\_dim=1」은 입력차원이 1이라는 뜻이며 입력 노드가 한개라고 생각하면 된다.

만약 x배열의 데이터가 2개라면 2, 3개라면 3으로 지정을 해준다.

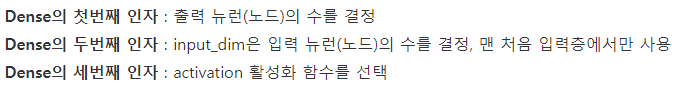
만든 시퀀스 오브젝트 model에 5개의 노드를 Dense레이어를 통해 연결해준다.

여기서 add를 통해 하나의 레이어를 추가해 주는 것이다.

Dense 레이어는 입력과 출력을 모두 연결해주며 입력과 출력을 각각 연결해 주는

가중치를 포함하고 있다. 입력이 3개 출력이 4개라면 가중치는 총 3X4인 12개가

존재하게 된다. Dense레이어는 머신러닝의 기본층으로 영상이나 서로 연속적으로 상관관계가 있는 데이터가 아니라면 Dense레이어를 통해 학습시킬 수 있는 데이터가 많다는 뜻이 된다.



**활성화 함수의 종류**

|  |  |
| --- | --- |
| relu | 은닉 층으로 학습  'relu' 는 은닉층으로 역전파를 통해 좋은 성능이 나오기 때문에  마지막 층이 아니고서야 거의 relu 를 이용한다. |
| sigmond | yes or no 와 같은 이진값을 반환한다. |
| softmax | 확률 값을 반환한다.  반환된 전체 합은 1이다. |

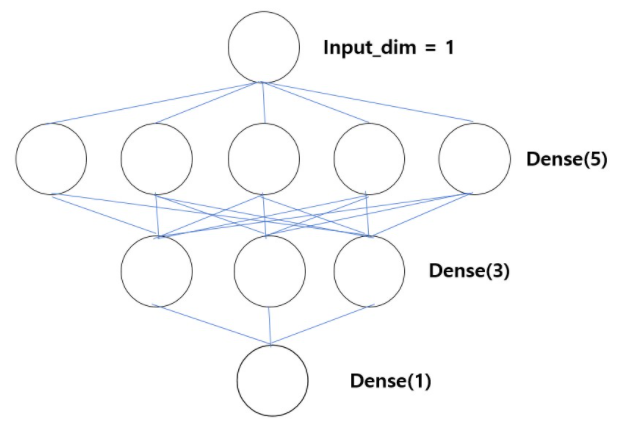
**model.add(Dense(3))**

두번째 레이어의 노드의 갯수는 3개

**model.add(Dense(1))**

마지막 레이어의 노드의 갯수는 출력 데이터 y의 갯수와 동일하게 1개로 지정

이렇게 만들어진 시퀀스 모델의 모양은 대략 이렇다.



**model.compile(loss='mse', optimizer='adam')**

만들어진 모델을 컴파일 한다.

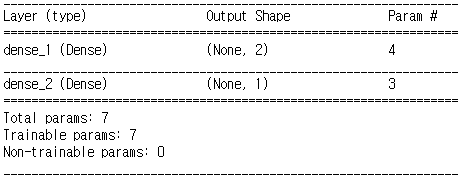
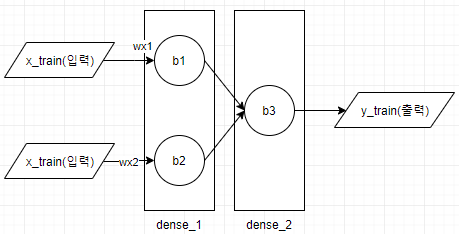
**model.fit(x, y, epochs=100, batch\_size=1)**

컴파일한 모델을 훈련을 시킨다.

다른예)

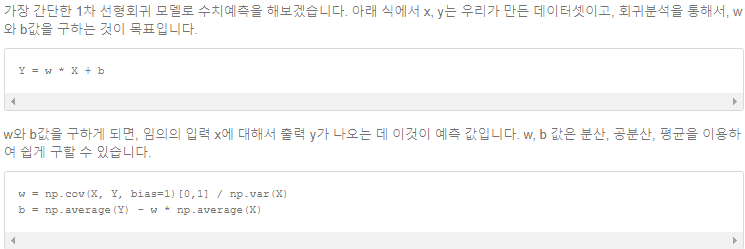
**model.add(Dense(2))**

**model.add(Dense(1))** #model.summary()

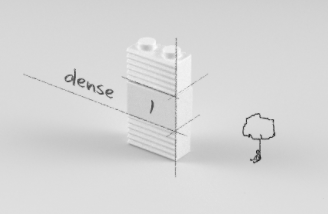




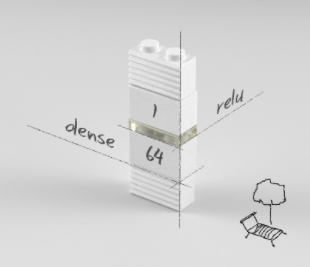
**선형회귀 모델**



**퍼셉트론 신경망 모델**



**다층퍼셉트론 신경망 모델**



**깊은 다층퍼셉트론 신경망 모델**

