강의 목표

1. 머신러닝의 기본 프로세스 – 가설 정의, 손실함수 정의, 최적화 정의
2. TensorFlow 2.0을 이용해서 선형회귀 함수를 구현해봅니다.

AI School.

머신러닝의 기본 프로세스 - 가설 정의, 손실함수 정의, 최적화 정의

- 모든 머신러닝 모델은 다음의 3가지 과정을 거칩니다. 이 과정은 앞으로 배우는 모든 머신러닝 알고리즘의 기본 토대이기 때문에 꼭 제대로 숙지하고 넘어가셔야 합니다.

① 학습하고자 하는 **가설**$_{\text{Hypothesis}}$ **h(θ)**을 수학적 표현식으로 나타낸다.

② 가설의 성능을 측정할 수 있는 **손실함수**$_{\text{Loss Function}}$ **J(θ)**을 정의한다.

③ 손실 함수 **J(θ)을 최소화**$_{\text{Minimize}}$ 할 수 있는 학습 알고리즘을 설계한다.

AI School.

1. 가설 정의

- 이 3가지 과정을 선형 회귀 모델에 대입해서 생각해보면 다음과 같습니다.
- 선형 회귀 모델은 선형 함수를 이용해서 회귀를 수행하는 기법입니다.

① 선형 회귀 함수는 학습하고자 하는 **가설**$_{\text{Hypothesis}}$ **h(θ)**을 아래와 같은 선형 함수 형태로 표현합니다.

$$y = Wx + b$$

- 이때 x와 y는 데이터로부터 주어지는 인풋 데이터, 타겟 데이터이고, W와 b는 **파라미터**$_{\text{Parameter}}$ θ라고 부르며 트레이닝 데이터로부터 학습을 통해 적절한 값을 찾아내야만 하는 값입니다.

AI School.

2. 손실 함수 정의

② 적절한 파라미터값을 알아내기 위해서는 현재 파라미터값이 우리가 풀고자 하는 목적$_{\text{Task}}$에 적합한 값인지를 측정할 수 있어야 합니다. 이를 위해 **손실 함수**$_{\textbf{Loss Function}}$ **J(θ)**를 정의합니다.

- 손실 함수는 여러가지 형태로 정의될 수 있습니다. 그 중 가장 대표적인 손실 함수 중 하나는 **평균제곱오차**$_{\textbf{Mean of Squared Error(MSE)}}$입니다. MSE는 다음 수식으로 정의됩니다.

$$MSE = \frac{1}{2n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

AI School.

2. 손실 함수 정의

- 예를 들어, 정답이 y=[1, 10, 13, 7]이고 우리의 모델의 예측값이 $\hat{y}$=[10, 3, 1, 4]와 같이 잘못된 값을 예측한다면 MSE 손실 함수는 아래와 같이 35.375라는 큰 값을 갖게 될 것입니다.

$$MSE = \frac{1}{2*4}\{(10-1)^2+(3-10)^2+(1-13)^2+(4-7)^2 = \mathbf{35.375}$$

- 하지만, 정답이 y=[1, 10, 13, 7]이고 우리의 모델의 예측값이 $\hat{y}$=[2, 10, 11, 6]와 같이 비슷한 값을 예측한다면 MSE 손실 함수는 아래와 같이 1.5라는 작은 값을 갖게 될 것입니다.

$$MSE = \frac{1}{2*4}\{(2-1)^2+(10-10)^2+(11-13)^2+(6-7)^2 = \mathbf{1.5}$$

- 이처럼 손실 함수는 우리가 풀고자 하는 목적에 가까운 형태로 파라미터가 최적화 되었을 때(즉, 모델이 잘 학습되었을 때) 더 작은 값을 갖는 특성을 가져야만 합니다. 이런 특징 때문에 손실 함수를 다른 말로 **비용 함수**$_{\text{Cost Function}}$라고도 부릅니다.

AI School.

3. 최적화 정의 – Gradient Descent

③ 마지막으로 손실 함수를 최소화하는 방향으로 파라미터들을 업데이트할 수 있는 학습 알고리즘을 설계해야 합니다.

- 머신러닝 모델은 보통 맨 처음에 랜덤한 값으로 파라미터를 초기화한 후에 파라미터를 적절한 값으로 계속해서 업데이트합니다.
- 이때, 파라미터를 적절한 값으로 업데이트하는 알고리즘을 **최적화**$_{\text{Optimization}}$ **기법**이라고 합니다. 여러 최적화 기법 중에서 대표적인 기법은 **경사하강법**$_{\text{Gradient Descent}}$입니다.
- 경사하강법은 현재 스텝의 파라미터에서 손실 함수의 미분값에 **러닝레이트 α**를 곱한만큼을 빼서 다음 스텝의 파라미터값으로 지정합니다.
- 따라서 손실 함수의 미분값이 크면 하나의 스텝에서 파라미터가 많이 업데이트되고 손실 함수의 미분값이 작으면 적게 업데이트 될 것입니다. 또한 러닝레이트 α가 크면 많이 업데이트, α가 작으면 적게 업데이트 될 것입니다.

3. 최적화 정의 – Gradient Descent

- 경사하강법의 파라미터 한 스텝 업데이트 과정을 수식으로 나타내면 다음과 같습니다.

$$\theta_i = \theta_i - \alpha \frac{\partial}{\partial \theta_i} J(\theta_0, \theta_1)$$

- 위 표현에서 θ는 우리가 학습하고자하는 파라미터를 나타냅니다. 선형회귀 모델에서 파라미터는 W와 b 입니다. 즉, θ=(b,W), θ_0 =b, θ_1 =W 입니다. W와 b 각각에 대해서 경사하강법 알고리즘의 한스텝 업데이트 과정을 풀어서 적어 보면 아래와 같습니다. 손실 함수로 오차제곱평균 MSE를 사용할 경우 θ_0 (=b), θ_1(=W)의 미분값은 아래와 같습니다.

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_0} \frac{1}{2n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 = \frac{\partial}{\partial \theta_0} \frac{1}{2n} \sum_{i=1}^{n} (W x_i + b - y_i)^2 = \frac{1}{n} \sum_{i=1}^{n} ((W x_i + b) x_i - y_i)$$

$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_1} \frac{1}{2n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 = \frac{\partial}{\partial \theta_1} \frac{1}{2n} \sum_{i=1}^{n} (W x_i + b - y_i)^2 = \frac{1}{n} \sum_{i=1}^{n} (W x_i + b - y_i) x_i$$

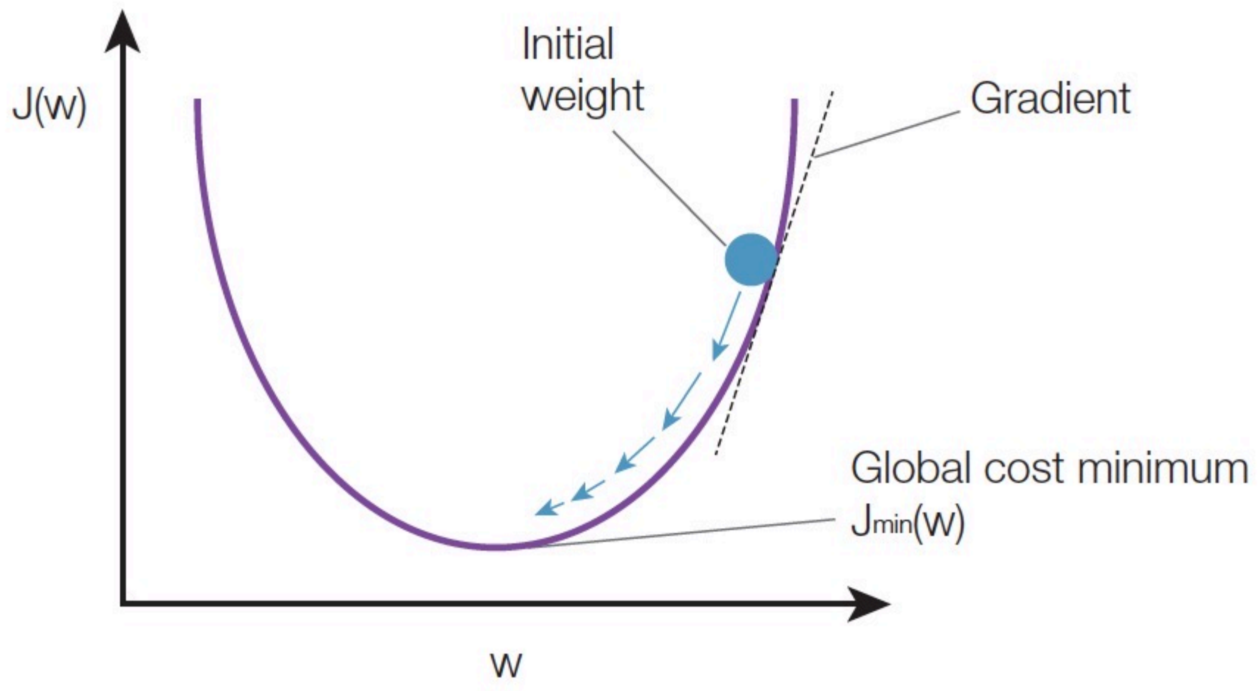
- 따라서 $\theta_0 = b, \theta_1 = W$의 경사하강법의 파라미터 한 스텝 업데이트는 다음과 같습니다.

$$\theta_0 = \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \theta_0 - \alpha \frac{1}{n} \sum_{i=1}^{n} \big((W x_i + b) x_i - y_i\big)$$

$$\theta_1 = \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \theta_1 - \alpha \frac{1}{n} \sum_{i=1}^{n} (W x_i + b - y_i) x_i$$

3. 최적화 정의 – Gradient Descent

- 아래 그림은 경사하강법의 동작 과정을 나타냅니다.



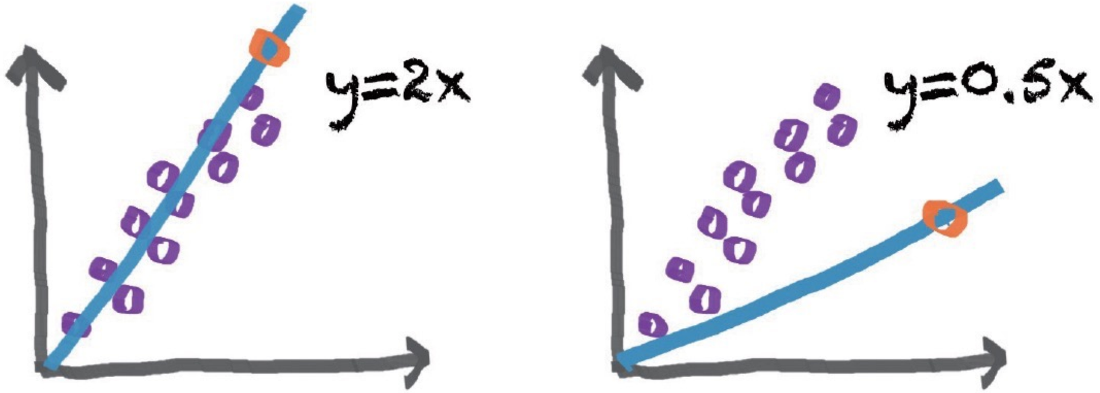
AI School.

머신러닝의 기본 프로세스 - 가설 정의, 손실함수 정의, 최적화 정의

- 경사하강법은 손실 함수가 최소가 되는 지점에서 종료하는 것이 가장 이상적이지만 현실적으로 언제 손실 함수가 최소가 될지 알기 어렵기 때문에 충분한 횟수라고 생각되는 횟수만큼 업데이트를 진행한 후 학습을 종료합니다.
- 이제 **머신러닝에 필요한 3가지 과정(모델 정의, 손실 함수 정의, 옵티마이저 정의)** 을 모두 살펴 보았습니다.

선형 회귀 (Linear Regression)

- 이제 머신러닝의 3가지 프로세스를 이용해서 선형 회귀 모델을 구현해봅시다. 아래 그림은 선형 회귀의 예시를 보여줍니다.

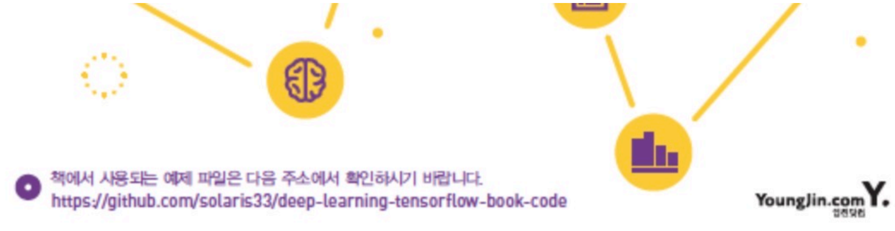


- 보라색 동그라미는 트레이닝 데이터, 파란색 라인은 선형 회귀 기법이 학습한 가설, 주황색 동그라미는 학습한 가설을 바탕으로 테스트 데이터에 대해 예측을 수행한 결과입니다.
- 왼쪽 그림은 선형 회귀 모델이 y=2x(W=2, b=0)로 가설을 학습한 경우, 오른쪽 그림은 선형 회귀 모델이 y=0.5x(W=0.5, b=0)로 가설을 학습한 경우입니다.
- 그림에서 볼 수 있듯이 보라색의 트레이닝 데이터는 y=2x 형태의 경향성을 띠고 있기 때문에 선형 회귀 모델이 잘 학습된 경우 y=2x 형태의 가설을 가지고 있어야 합니다. 만약 잘못된 선형 함수가 학습된 경우 오른쪽 그림과 같이 테스트 데이터에 대해 부정확한 예측값을 출력하게 될 것입니다.

AI School.

TensorFlow 2.0을 이용한 선형회귀 알고리즘 구현

- 선형 회귀를 알고리즘을 TensorFlow 2.0 코드로 구현해봅시다.
- https://github.com/solaris33/deep-learning-tensorflow-book-code/blob/master/Ch03-TensorFlow_Basic/3.3-linear_regression_v2.py



Chapter 2 - 텐서플로우 소개

- 텐서플로우 설치 체크 (Code) (TF v2 Code)

Chapter 3 - 텐서플로우 기초와 텐서보드

- 텐서플로우 기초 – 그래프 생성과 그래프 실행 (Code) (TF v2 Code)
- 플레이스홀더 (Code) (TF v2 Code)
- 선형 회귀(Linear Regression) 알고리즘 (Code) (TF v2 Code)
- 선형 회귀(Linear Regression) 알고리즘 + 텐서보드(TensorBoard) (Code) (TF v2 Code) (TF v2 Keras Code)

AI School.

Thank you!