

소프트웨어 공학개론

1조 프로젝트 발표

Cat/Dog Classifier



SWE3002_41

Team 1

강동준, 강영호, 박상민, 이선종, 조민구

SungKyunKwan University

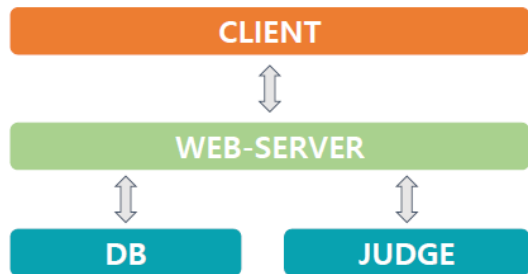
Mar. 26, 2022

계획했던 내용

Plan



Development Direction



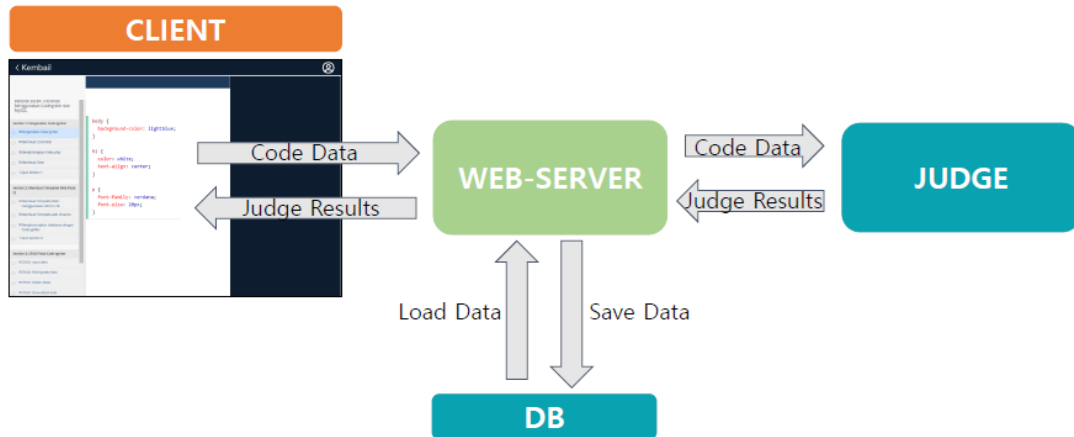
Back-End

Web Server (Python Django)

1. 로그인 관리
2. 사용자의 요청시 Judge 프로세스와 통신
3. DB 통신

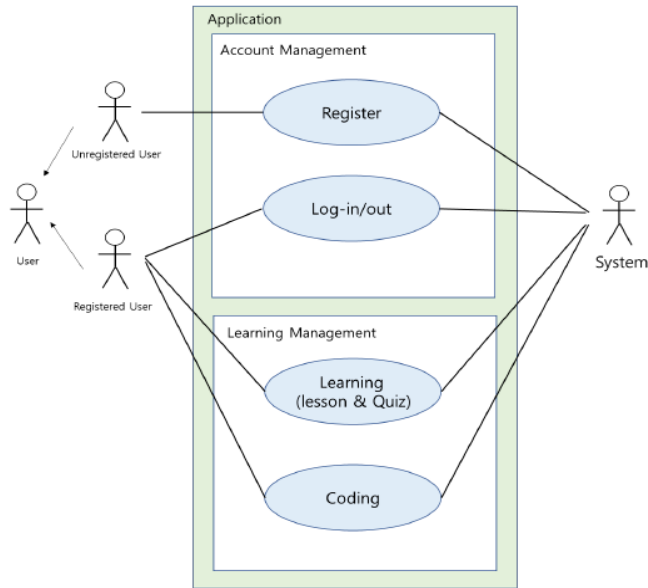
DB (SQLite)

1. User 로그인 정보
2. 교육 진행도 / 달성률

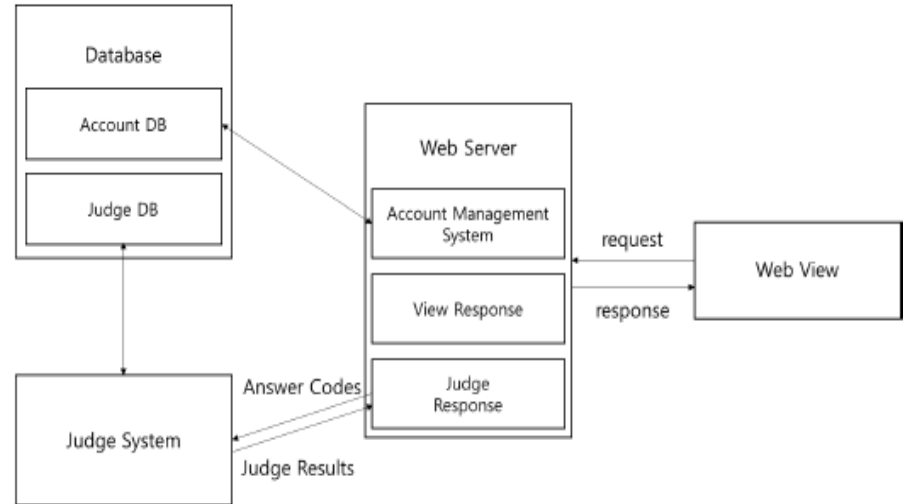


Requirement Specification

Use case Diagram



System Architecture



Changes

- 대학교 이메일을 통한 본인 인증 -> 없음
// 구현 상의 문제
- 교수가 학생들의 수업 진행 현황 및 점수 확인 -> 없음
- 학습 대상 : 성균관대학교 학생 -> 모든 사용자
- 로그인 및 회원가입 : 학번이 ID -> 이메일이 ID
// 위 변경사항으로 인한 자연스러운 변경점
- 강의 영상 -> 없음
- 커스터마이징된 관리자 페이지 -> django 기본 관리자 페이지
- DB에 Problem을 저장 -> Problem DB 미구현
// DB없는 쪽이 더 효율적
// 시간과 비용에 의한 선택

Teaching Flow Direction



Teaching Flow

Teaching Flow는 전체 Cat/Dog Classifier를 처음부터 끝까지 학생이 학습하고 이미지 classifier를 학습 및 구현하는 것이 목표

다음과 같은 8개의 챕터로 나누어 전체 머신러닝 학습 클래스를 구성

- pytorch Library
- Tensor Manipulation
- Gradient Descent Algorithm
- Data Processing
- Convolution Neural Network (CNN)
- Training Process
- Validation Process
- Test Process

각각의 챕터는 이론과 함께 학습자의 이해도를 높이기 위해 이론을 공부한 후 간단한 퀴즈 및 실습을 통해 직접 문제를 해결하도록 구성

Teaching Flow Example

선형 회귀(Linear Regression)이란 학습 데이터와 가장 잘 맞는 하나의 직선을 찾는 일입니다.

선형 회귀의 가설(직선의 방정식)은 아래와 같은 형식을 가집니다.

$y = Wx + b$, x 와 곱해지는 W 를 가중치(Weight)라고 하며, b 를 편향(bias)라고 합니다.

어떤 데이터를 이 가설에 대입하면 $H(x) = Wx + b$ 라는 식으로 표현됩니다.

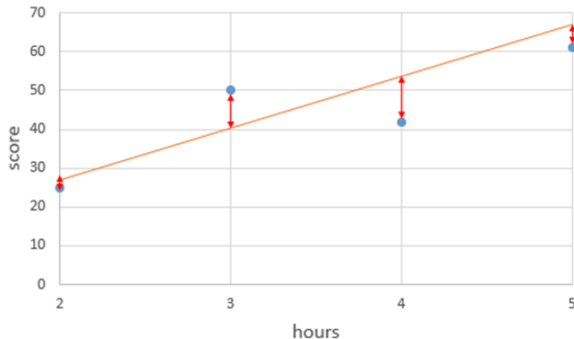
데이터와 직선이 얼마나 잘 맞는지에 대해 수학적인 방법으로 알 수 있습니다.

비용 함수(cost function) = 손실 함수(loss function) = 오차 함수(error function) = 목적 함수(objective function)

보통 비용 함수와 손실 함수라고 부릅니다.

n 개의 (x, y) 의 학습데이터가 있는 상황을 가정했을 때 모든 데이터들은 임의의 직선과 차이인 오차를 가지고 있고, 음수와 양수 모두 존재하기 때문에 수학적인 방법을 통해서 Cost function을 표현해줘야 합니다. 이 값을 평균 제곱 오차(Mean Squared Error)로 정리해주면 아래와 같은 식으로 표현됩니다.

단순하게 생각해보면, cost function의 값이 가장 낮은 직선이 학습데이터와 가장 잘 맞는 직선이라 할 수 있을 것입니다.



Teaching Flow Example

아래와 같이, Hypothesis 와 Cost Function 을 설정해 보고 출력해보세요

```
import torch

#주석처리부분이 사용자가 구현해야 할 부분

def example2_1():

    # x_train = torch.FloatTensor([[1], [2], [3]])

    # y_train = torch.FloatTensor([[2], [4], [6]])

    # # 가중치 w 를 0 으로 초기화하고 학습을 통해 값이 변경되는 변수임을 명시합니다

    # W=torch.zeros(1, requires_grad=True)

    # # 편향 b 를 0 으로 초기화하고 학습을 통해 값이 변경되는 변수임을 명시합니다

    # b=torch.zeros(1, requires_grad=True)

    # # H(x)

    # hypothesis = x_train * W + b

    # # cost

    # cost = torch.mean((hypothesis - y_train) ** 2 )

    # print(W, b)

    # print(hypothesis)

    # print(cost)

    # return hypothesis, cost

if __name__=="__main__":

    hypothesis,cost=example2_1()
```

answer

```
tensor([0.], requires_grad=True) tensor([0.], requires_grad=True)

tensor([[0.],

        [0.],

        [0.]], grad_fn=<AddBackward0>)

tensor(18.6667, grad_fn=<MeanBackward0>)
```

Differentiation

1. Python 설치 없이 웹사이트에 접속하기만 하면 Python Code를 짜볼 수 있고 실행 결과를 확인할 수 있다.

2. Classifier를 만들기 위한 pytorch에 관한 기본 개념들에 관한 문제를 직접 코딩하면서 풀어볼 수 있고, 정확한 채점결과를 볼 수 있다.



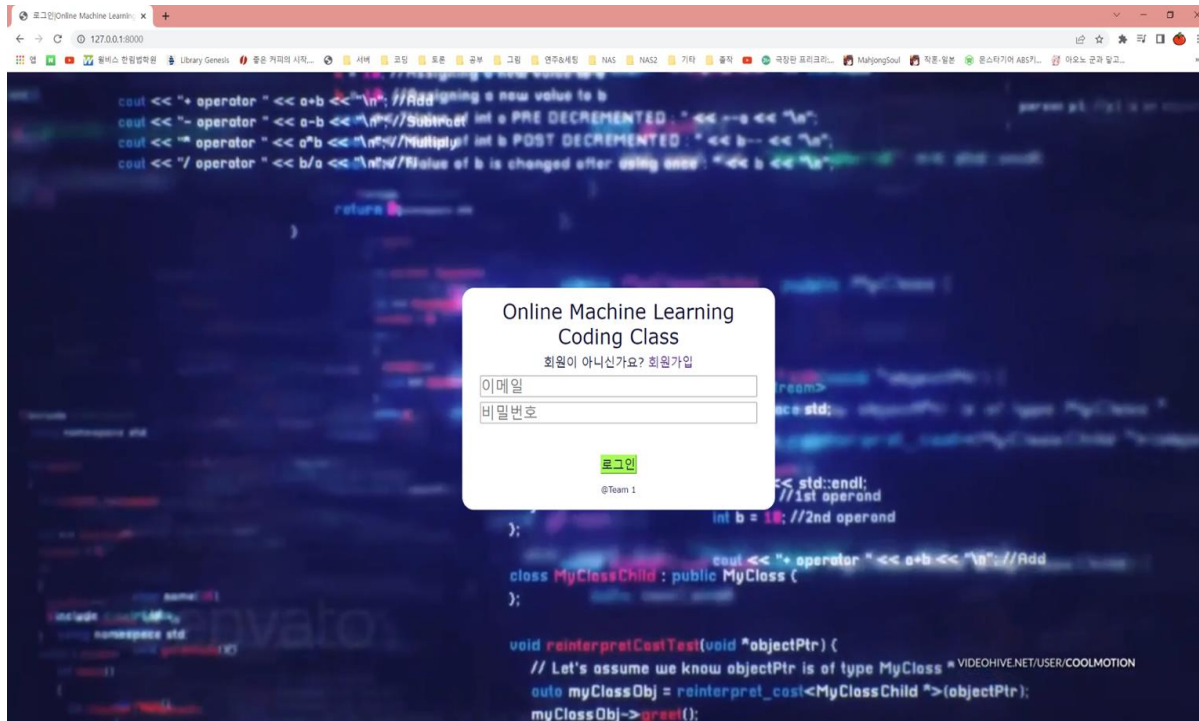
구현 결과

Development



로그인 시스템

- 로그인 시 세션 ID 발급
- 세션ID 있을 시 로그인 생략하고 강의 페이지로 점프



회원이가입 페이지

회원이가입

- 이메일, 이름, 비밀번호 세가지 필드
- 이메일 작성 시 중복검사 실행
- 제출시 이메일 유효성, 중복검사, 비밀번호 확인

Online Machine Learning Coding Class

회원인가요? [로그인](#)

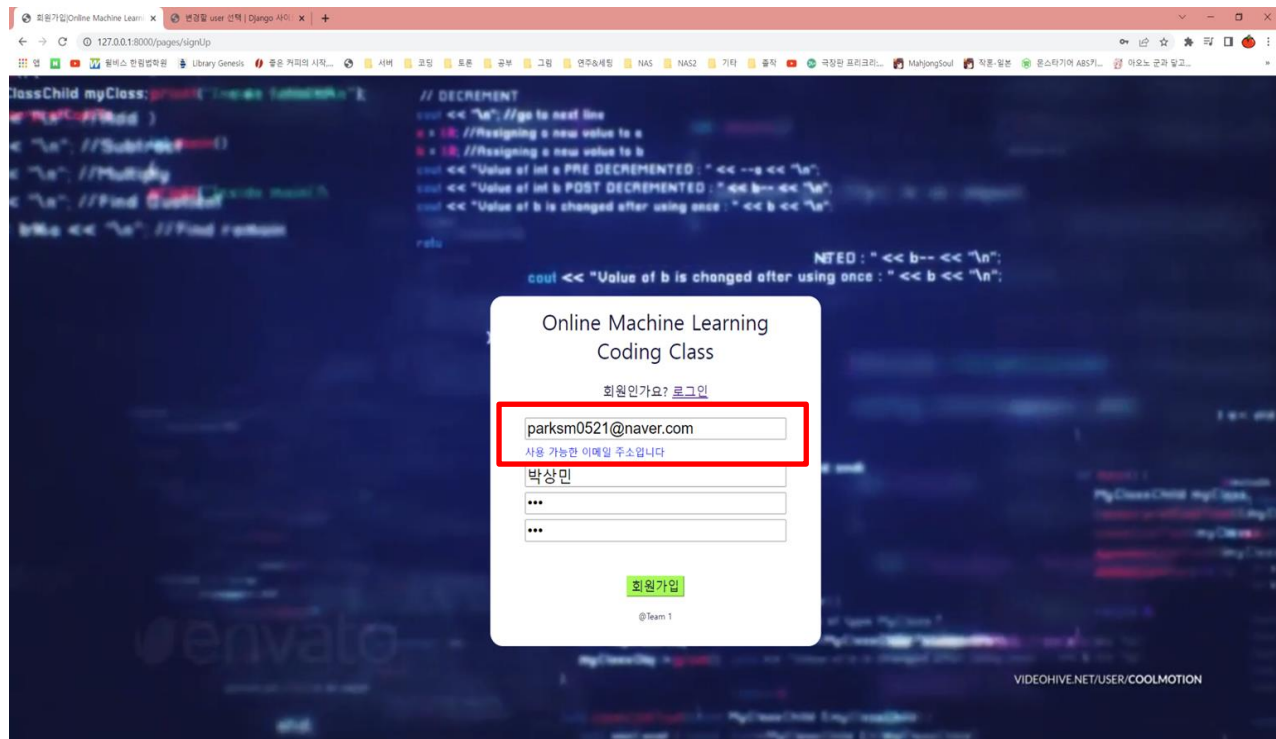
이미 존재하는 이메일 주소입니다

© Team 1

회원이가입 페이지

회원이가입

- 이메일, 이름, 비밀번호 세가지 필드
- 이메일 작성 시 중복검사 실행
- 제출시 이메일 유효성, 중복검사, 비밀번호 확인



홈페이지/수업

사이드바 & 버튼

- 다음 수업내용
으로 이동



홈페이지/수업

탭 네비게이션

온라인 머신러닝 수업 ->

홈페이지/수업으로 이동

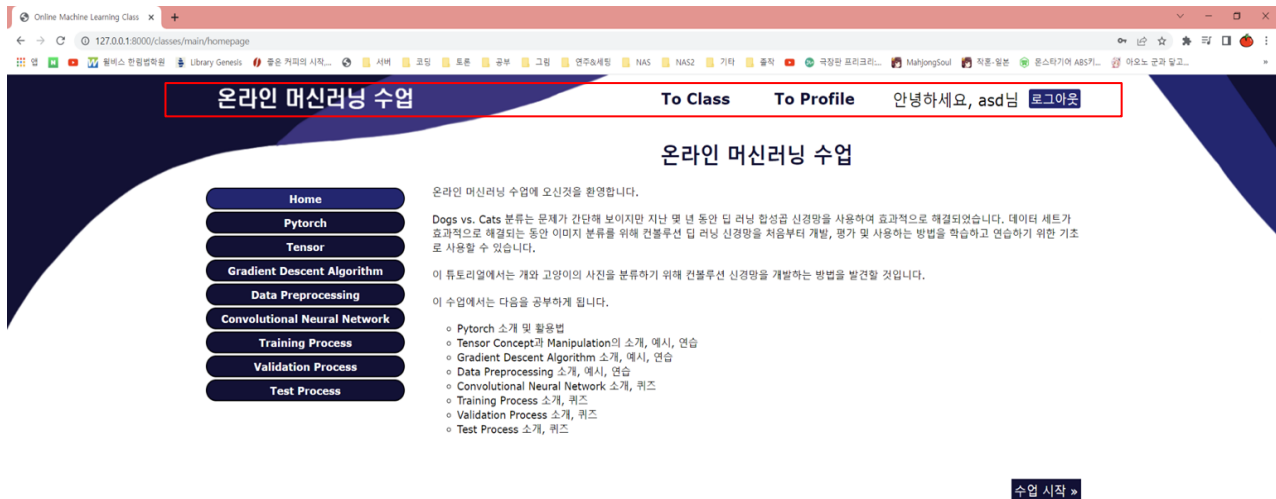
To Class ->

실습창에 있을때 수업창으로 이동

To Profile ->

프로필화면으로 이동

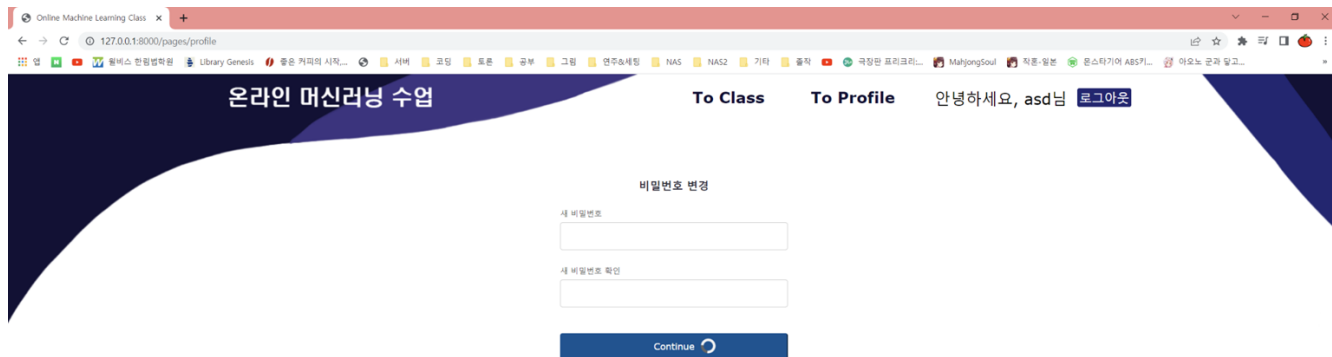
로그아웃 -> 로그아웃



비밀번호 변경

비밀번호와 비밀번호
확인을 통해 검증,

비밀번호 변경 요청을
서버로 보내는 역할



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:8000/pages/profile'. The website has a dark blue header with the text '온라인 머신러닝 수업' (Online Machine Learning Class) and navigation links 'To Class' and 'To Profile'. A user is logged in as 'asd님' (asd님). The main content area is titled '비밀번호 변경' (Change Password) and contains two input fields: '새 비밀번호' (New Password) and '새 비밀번호 확인' (Confirm New Password). Below the fields is a blue button labeled 'Continue' with a circular arrow icon.

각 챕터의 학습내용에
대한 문서가 작성되어
있다.

온라인 머신러닝 수업

Home

Pytorch

Tensor

Gradient Descent Algorithm

Linear Regression: Class

Linear Regression: Example

GD in Linear Regression: Class

GD in Linear Regression: Example

GD in Linear Regression: Practice

Data Preprocessing

Convolutional Neural Network

Training Process

Validation Process

Test Process

To Class

To Profile

안녕하세요, asd님

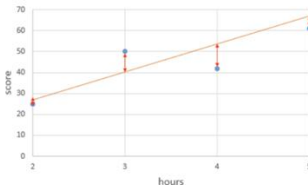
로그아웃

Gradient Descent Algorithm

Linear Regression

선형 회귀(Linear Regression)이란 학습 데이터와 가장 잘 맞는 하나의 직선을 찾는 일입니다. 선형 회귀의 가설(직선의 방정식)은 아래와 같은 형식을 가집니다.

$$y = Wx + b$$
$$x$$
와 y 의 공제지는 W 를 가중치(Weight)라고 하며, b 를 편향(bias)라고 합니다. 어떤 데이터를 이 가설에 대입하면 $H(x) = Wx + b$ 라는 식으로 표현됩니다.



데이터와 직선이 얼마나 잘 맞는지에 대해 수리적인 방법으로 알 수 있습니다.

비용 함수(cost function) = 손실 함수(loss function) = 오차 함수(error function) = 목적 함수(objective function) 보통 비용 함수와 손실 함수라고 부릅니다.

n 개의 (x, y) 의 학습데이터가 있는 상황을 가정했을 때 모든 데이터들은 임의의 직선과 차이인 오차를 가지고 있고, 음수와 양수 모두 존재하기 때문에 수리적인 방법을 통해서 Cost function을 표현해줘야 합니다.

이 값을 평균 제곱 오차(Mean Squared Error)로 정리해주면 아래와 같은 식으로 표현됩니다.

$$cost(W, b) = \frac{1}{n} \sum_{i=1}^n [y^{(i)} - H(x^{(i)})]^2$$

단순하게 생각해보면, cost function의 값이 가장 낮은 직선이 학습데이터와 가장 잘 맞는 직선이라 할 수 있을 것입니다.

18

Example 페이지

Class 페이지에서 학습한 내용에 대한 예제코드를 제공

출력버튼을 통해 결과를 확인할 수 있고 오류 발생 시 Log칸을 통해 확인 가능

Example 페이지

Class 페이지에서 학습한 내용에 대한 예제코드를 제공

출력버튼을 통해 결과를 확인할 수 있고 오류 발생 시 Log칸을 통해 확인 가능

Example 페이지

Class 페이지에서 학습한 내용에 대한 예제코드를 제공

출력버튼을 통해 결과를 확인할 수 있고 오류 발생 시 Log칸을 통해 확인 가능

온라인 머신러닝 수업

To Class To Profile 안녕하세요, asd님 로그인

Understanding of Tensor

Example
아래와 같이, Hypothesis와 Cost Function을 설정해 보고 출력해보세요

Input

```
1 x_train = torch.FloatTensor([[1], [2], [3]])
2 y_train = torch.FloatTensor([[3], [4], [6]])
3 # 가중치 W를 0으로 초기화하고 학습을 통해 값이 변경되는 변수임을 명시합니다
4 W=torch.zeros(1, requires_grad=True)
5 # 편향 b를 0으로 초기화하고 학습을 통해 값이 변경되는 변수임을 명시합니다
6 b=torch.zeros(1, requires_grad=True)
7 # H()
8 hypothesis = x_train * W + b
9 # cost
10 cost = torch.mean((hypothesis - y_train) ** 2)
11
12 print('hypothesis : ', hypothesis)
13 print('cost : ', cost)
14 print('W : ', W)
15 print('b : ', b)
16
```

Output

출력

Log

```
1 File "<string>". line 16
2
3
4 SyntaxError: unexpected EOF while parsing
```

« 전 수업 다음 수업 »

Practice 페이지

Class 페이지에서 학습한 내용에 대한 실습문제를 제공

제출 시 서버는 작성한 코드의 정오를 파악해서 돌려주고 Example과 같이 out, log를 받음

The screenshot displays the 'Understanding of Tensor' practice page. The sidebar on the left contains the following navigation links: Home, Pytorch, Tensor, Gradient Descent Algorithm, Linear Regression: Class, Linear Regression: Example, GD in Linear Regression: Class, GD in Linear Regression: Example, GD in Linear Regression: Practice, Data Preprocessing, Convolutional Neural Network, Training Process, Validation Process, and Test Process. The main content area features a code editor with the following Python code:

```
def practice_2_2():
    # 데이터
    x_train = torch.FloatTensor([[25], [50], [42], [61], [81]])
    y_train = torch.FloatTensor([181, [159], [130], [180], [244]])
    # 모델 초기화
    W = torch.zeros(1, requires_grad=True)
    b = torch.zeros(1, requires_grad=True)
    # optimizer 설정
    optimizer = torch.optim.SGD([W, b], lr=0.0001)
    nb_epochs = 100 # 원하는만큼 경사 하강법을 반복
    for epoch in range(nb_epochs + 1):
        # H(x) 계산
        hypothesis = x_train * W + b
        # cost 계산
        cost = torch.mean((hypothesis - y_train) ** 2)
        # cost로 H(x) 계산
        optimizer.zero_grad()
        cost.backward()
        optimizer.step()
        # 10에이더를 출력
        if epoch % 10 == 0:
            print('Epoch {:.4d}/{} W: {:.3f}, b: {:.3f} Cost: {:.6f}'.format(
                epoch, nb_epochs, W.item(), b.item(), cost.item()))
    return hypothesis, cost, W, b
```

Below the code editor, there is an 'Output' section and a 'Log' section. At the bottom of the page, there are buttons for '전 수업' (Previous Lesson) and '다음 수업' (Next Lesson).

Practice 페이지

Class 페이지에서 학습한 내용에 대한 실습문제를 제공

제출 시 서버는 작성한 코드의 정오를 파악해서 돌려주고 Example과 같이 out, log를 받음

The screenshot displays the 'Practice' page of an online machine learning class. The sidebar on the left contains navigation links: Home, Pytorch, Tensor, Gradient Descent Algorithm, Linear Regression: Class, Linear Regression: Example, GD in Linear Regression: Class, GD in Linear Regression: Example, GD in Linear Regression: Practice, Data Preprocessing, Convolutional Neural Network, Training Process, Validation Process, and Test Process. The main content area shows a question about linear regression and a code editor with a PyTorch implementation of gradient descent. The output shows the training progress over 100 epochs.

Question: 위 Data에 대해 Linear Regression 방식으로 Hypothesis와 Cost 식을 만들고 Gradient Descent를 적용하여 적절한 Learning Rate를 설정해보고, Cost < 30을 만족하게 하는 W와 b를 찾아보세요.

Answer

Input

```
def practice_2_2():
    # 데이터
    x_train = torch.FloatTensor([[25], [50], [42], [61], [80]])
    y_train = torch.FloatTensor([[81], [159], [130], [180], [244]])
    # 모델 초기화
    W = torch.zeros(1, requires_grad=True)
    b = torch.zeros(1, requires_grad=True)
    # optimizer 설정
    optimizer = torch.optim.SGD([W, b], lr=0.0001)
    nb_epochs = 100 # 원하는만큼 경시 횟수를 반복
    for epoch in range(nb_epochs + 1):
        # H(x) 계산
        hypothesis = x_train * W + b
        # cost 계산
        cost = torch.mean((hypothesis - y_train) ** 2)
        # cost로 H(x) 갱신
        optimizer.zero_grad()
        cost.backward()
        optimizer.step()
        # 10에이더리프 로그 출력
        if epoch % 10 == 0:
            print('Epoch {:4d}/! W: {:.3f}, b: {:.3f} Cost: {:.6f}'.format(
                epoch, nb_epochs, W.item(), b.item(), cost.item()))
    return hypothesis, cost, W, b
```

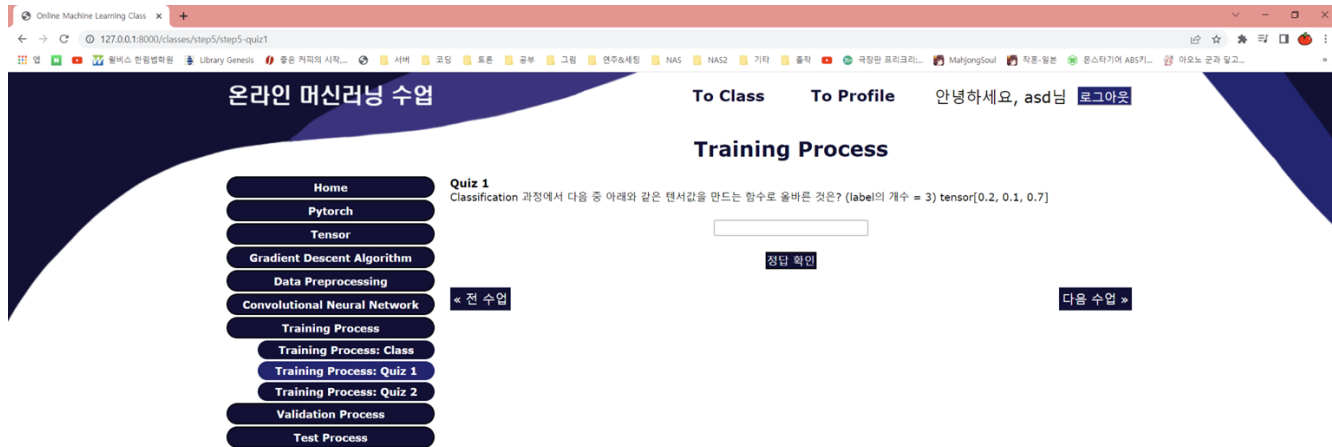
Output

```
Epoch 0/100 W: 1.837, b: 0.032 Cost: 28135.598609
Epoch 10/100 W: 3.059, b: 0.055 Cost: 20.590923
Epoch 20/100 W: 3.059, b: 0.056 Cost: 20.587661
Epoch 30/100 W: 3.059, b: 0.058 Cost: 20.584396
Epoch 40/100 W: 3.059, b: 0.060 Cost: 20.581274
Epoch 50/100 W: 3.059, b: 0.062 Cost: 20.578161
Epoch 60/100 W: 3.059, b: 0.063 Cost: 20.575052
Epoch 70/100 W: 3.059, b: 0.065 Cost: 20.571987
Epoch 80/100 W: 3.059, b: 0.067 Cost: 20.568867
Epoch 90/100 W: 3.059, b: 0.069 Cost: 20.565747
```

Quiz 페이지

Class 페이지에서 학습한 내용에 대한 퀴즈를 제공한다.

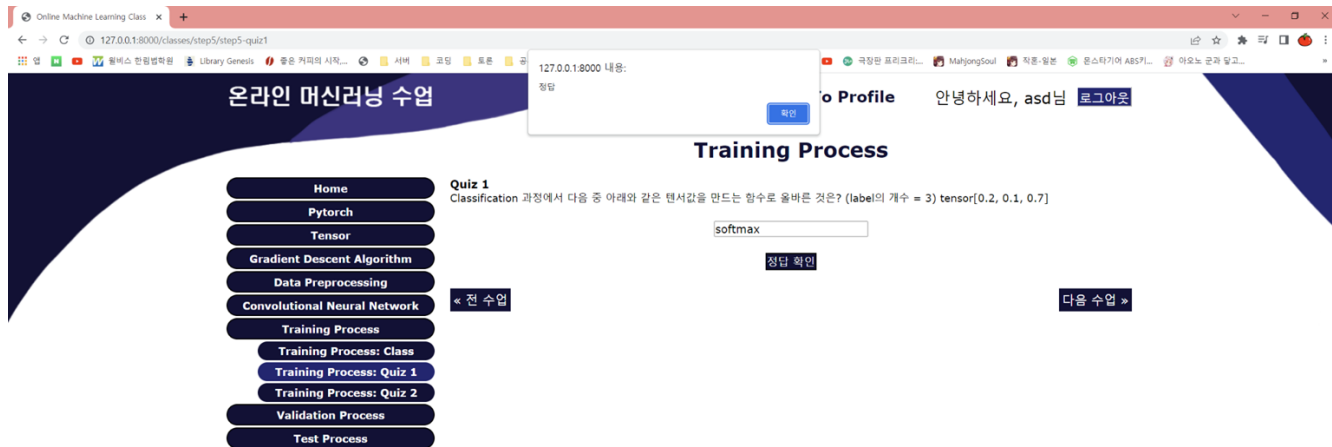
제출 시 서버는 문자열 답안의 정오를 파악해서 돌려준다



Quiz 페이지

Class 페이지에서 학습한 내용에 대한 퀴즈를 제공한다.

제출 시 서버는 문자열 답안의 정오를 파악해서 돌려준다



Test Learning Process (예정)

1. 사용자가 모든 내용을 학습한 이후 학습한 내용을 바탕으로 전체 ML 코드를 작성한다.
2. ML코드는 서버로 제출되고 서버에서는 ML코드로 학습을 진행한다.
3. 학습한 AI가 개와 고양이 사진을 판별하고 정확도를 사용자에게 반환한다.
4. 사용자는 최종적으로 자신이 작성한 ML코드의 정확도를 확인할 수 있다.

References

[Teaching Flow] : Pytorch로 딥러닝 입문 (wicidocs.net/book/2778)
모두를 위한 딥러닝 시즌2 pytorch

[Frontend]: html Documentation(<https://devdocs.io/html/>)
CSS Documentation(<https://devdocs.io/css/>)
jquery Docuemtnation(<https://api.jquery.com/>)
Background Video(<https://youtu.be/RR2EI8EEOWw>)

[Backend] : Django Documentation(<https://docs.djangoproject.com/en/4.0/>)
Python Documentation[subprocess]
(<https://docs.python.org/ko/3/library/subprocess.html>)

감사합니다

@Team1