연구 내용 요약 및 계획

|  |  |
| --- | --- |
| 날짜 | 2020년 01월 31일 |
| **1. 이 번주 연구 관련 학습 내용 정리**   1. 딥러닝 : 텐서플로우를 이용하여 regression알고리즘 학습 2. 도커 : 도커란 무엇이며 언제 쓰이고 어떻게 사용하는지 학습   \*지난주에 못 올린거 포함하여 같이 작성함. | |
| **2. 다음 주 학습 계획**  - 텐서플로우를 이용한 딥러닝 강의(강의 내의 프로젝트 진행하여 결과물 산출)  <https://www.edwith.org/boostcourse-dl-tensorflow/lecture/41847> | |
| 1. **이번주 학습 내용 정리**   **Docker는** 서로 다른 환경을 사용할 경우 도커에 개발 환경을 구성해두고 그 환경을 여러 사람이 나눠쓰기 편하게 해주며, 하나의 서버를 이용하여 여러 개의 서버를 사용할 수 있다.  Docker를 통해 tensorflow로 regression방법을 통한 딥러닝을 진행해봤다.  regression이란 데이터 들의 평균을 구하는 방법이며 line regreesion은 데이터들과 가장 근사한 직선의 방정식이다.  최소비용을 구하기 위해 사용하는데 비용이 적을수록 원하는 값을 잘 찾은것이다.  비용은 가상의 값과 실제 데이터의 값을 뺸 값이다.  다른 방법으로는 최소비용을 찾기 위한 방법으로는 경사하강법이 있다.  경사하강법은 비용의 그래프의 접선의 그래프들 중 기울기가 0이 되는 직선의 방정식이 최소비용을 나타낸다라는 것이다. | |

20.01.20

Optimization model 는 목적함수를 가지고 있는데 이는 문제를 해결하는 방법이며 옵션들을 가지고 있다. 목적함수는 최대화 하거나 최소화하는 것이며 제한조건을 갖는다.

예) 냅색문제(배낭문제) = 배낭의 양이 한정되어 있을 때 최고의 이익을 내도록 물건을 담는 방법.

0/1냅색문제, = 물건을 가져가거나 못가져가거나

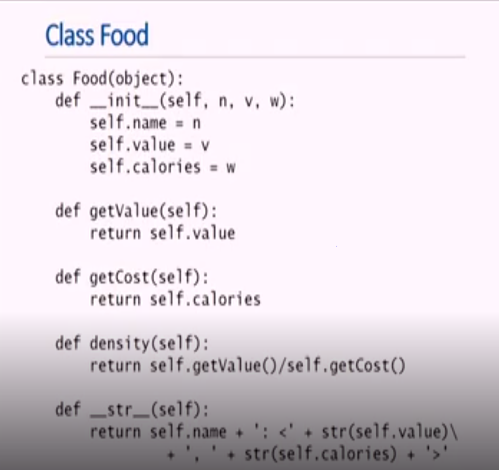
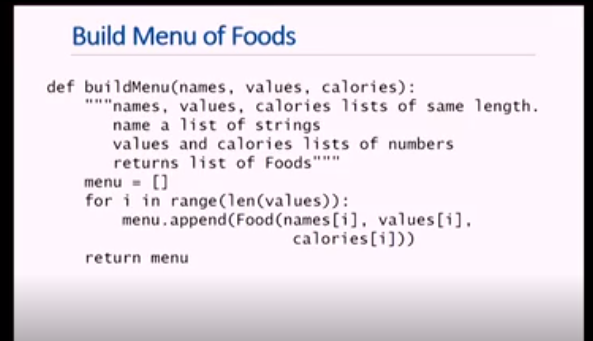
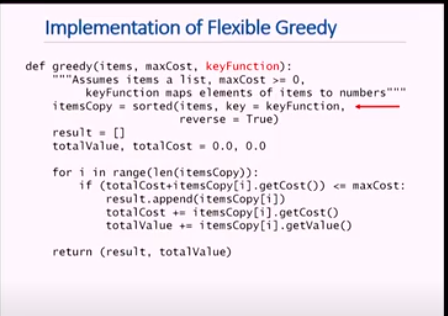
연속냅색문제 = 일부만을 배낭에 담아 가져갈수 있다.(탐욕 알고리즘)

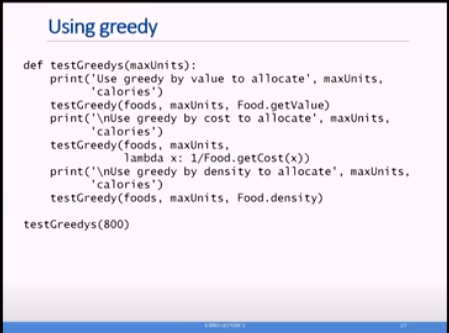
0/1배낭문제

최대무게 : w, 각각의 item : <value, weight>, vector:L(가능한 물건)., length : n

V[i] = 0/1( 물건을 가져가고 말고를 표시) I[i] = 물건의 무게

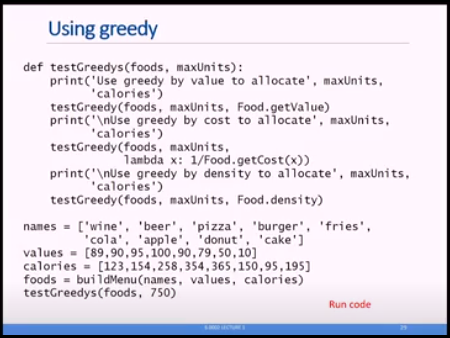
:정확한 해를 구하기 어려움

Keyfuntion = 물건의 원소를 숫자에 대응시켜주는 일을 한다

모든 물건을 정렬 (= 팀정렬 사용. 시간복잡도는 nlogn.) + for 루프(n번 반복 ) = o(nlogn)

람다 = 익명함수를 만들떄 사용. (함수를 한줄에 쓸수 있는경우 유용. 아닐경우 디버깅이 어려움)

Lambda<id1, id2…idn> : <원하는 식>’

Lambda x =? x의 자료형은 Food

탐욕알고리즘의 장점

* 구현이 쉽다
* 빠르다.

탐욕알고리즘의 단점

* 실제로 문제를 푸는 것이 아닌 최적화를 하는 것

탐색트리(search tree)

= 오른쪽은 취했을때의 결과, 왼쪽은 취하지 않았을때의 결과를 자식노드에

20.01.21

<https://www.edwith.org/boostcourse-dl-tensorflow/lecture/42993/>

도커 설치 및 개발 환경 구축

* 도커는 리눅스 환경에서 사용가능. 원도우즈에서 사용하기 위해 가상환경에서 사용함
* Virtualization = enable로 바꿔준 후 docker tolbox설치

Do0xker toolbox설치 시 bash.exe.가 없다는 오류 발생 -> 깃 배쉬 설치 후 경로 설정

<https://sukill.tistory.com/26>

error during connect: Get http://%2F%2F.%2Fpipe%2Fdocker\_engine/v1.40/version: open //./pipe/docker\_engine: The system cannot find the file specified. In the default daemon configuration on Windows, the docker client must be run elevated to connect. This error may also indicate that the docker daemon is not running.

이러한 에러 발생. -> 여러ㅜ방법 시도했으나 실패

20.01.22

<https://www.youtube.com/watch?v=chnCcGCTyBg>

docker가 왜 좋은지 5분안에 설명해줌-노마드코더

서로 다른 환경을 사용할 경우 도커에 개발 환경을 구성해두고 그 환경을 여러 서버에서 가져다 사용가능.

원하는 개발 환경을 파일에 저장하면, docker는 이를 원하는 어떤 머신에든 해당 환경을 시뮬레이션 해준다.

이러한 환경들은 각기 독립적으로 존재하기 때문에 원하는 무슨 환경이든 모듈식으로 관리 가능.

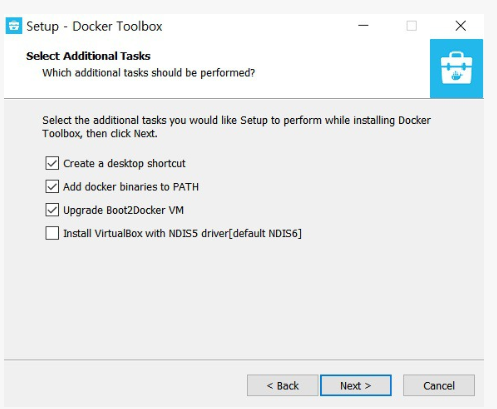
* 서버들을 추가 구매가 필요없다.

<https://www.youtube.com/watch?v=tPjpcsgxgWc> docker가 뭐고 왜 쓰는건가요-얄팍한 코딩사전

20.01.27

노트북에 도커 새로 설치(docker toolbox)

설치시 chocoletely사용하여 설치함. 설치시 virtualbox가 필요하다고 하여 virtualbox최신버전을 설치-> 하지만 실행이 안됨

virtualbox버전을 5.xx로 바꿔서 설치-> 실행은 되는 중간에 virtualbox가 꺼져서 실행이 된다는 에러 발생 -> 모두 지우고 다시 공홈에서 설치

공홈에서 깃으로 가서 .exe파일을 설치->초기 설치 시 사진의 제일 마지막 체크박스 체크 필요

-> 이를 통해 설치 완료

설치 후 개발에 필요한 이미지 다운로드 <https://github.com/deeplearningzerotoall/TensorFlow/blob/master/docker_user_guide.md>

20.01.28

ML = Machine Learning

* Limitations of explicit programming. 자료나 현상에서 자동으로 배우도록 하는 것
* Supervised/unsupervised = if labeled examples

Regression : wide range result ex) 0~100 score

Binary classification : choice between A and B ex) pass/nonpass

Multi-label classification : have more three label to result ex) A, B, C, D, F score

Simple linear regression

Regression : 퇴보, 후퇴, 되돌아간다(x) => regressiong toward the mean = 전체 평균으로 되돌아간다

Linear regression : 데이터를 가장 잘 대변하는 직선의 방정식의 기울기와 y절편을 찾는 것

Hypothesis H(x) = Wx + b;

Cost는 H(x)와 실제 데이터와의 차이(뺀 값).(=loss=error)

Cosr(w) = H(x)와 실제 데이터를 뺸 값의 제곱의 평균.(Wxi-yi)^2/m

Minimize cost(W, b)

Tf.reduce\_mean() = 평균을 구함(차원을 하나 줄인다는 의미에서 reduce)

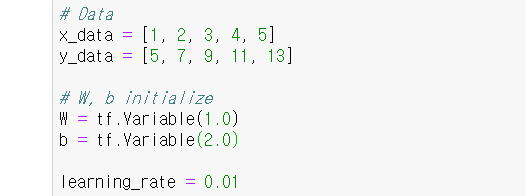
20.01.29

Gradient descent = 경사, 기울기 아래로,하강 =>경사하강법 : cost가 minimize되는 w,b를 찾는 알고리즘

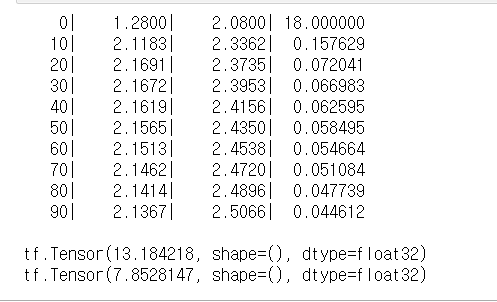
With GradientTape() as tape :

Assign\_sub = -=

Learning rate : 구한 값을 얼마만큼 반영할지를 결정하는 값. 주로 0.0001 0.00001을 사용



데이터 셋을 2x+3으로 하고 알고리즘에 적용해본 결과(100번 실행하였다)

 실행횟수 w값 b값 cost

w값은 2로수렴하고 b값은 3으로 수렴하는 중이다. 그러므로 cost값이 0에 가까워지는 것을 알수 있다.

Simplified hypothesis : H(x) = Wx [Cost : cost(w)]

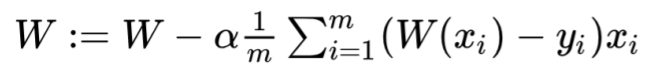
Cost function : 항상 이차함수의 그래프를 그린다. 그 중 최저점이 minimize cost가 된다.

=>gradient descent alforith사용(cost가 최소화되는 값을 찾는 알고리즘)

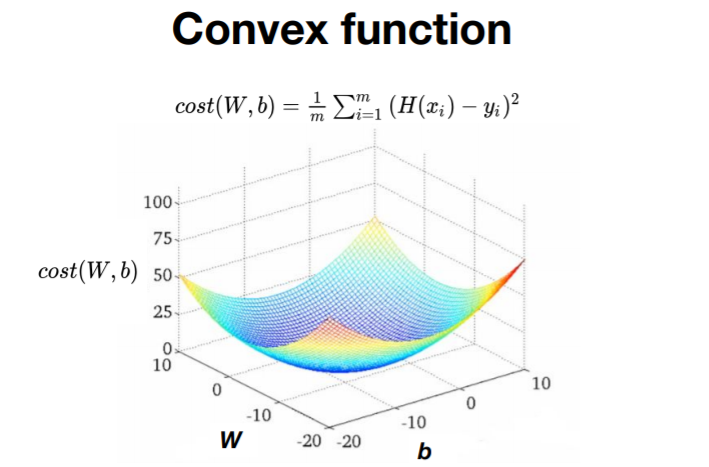
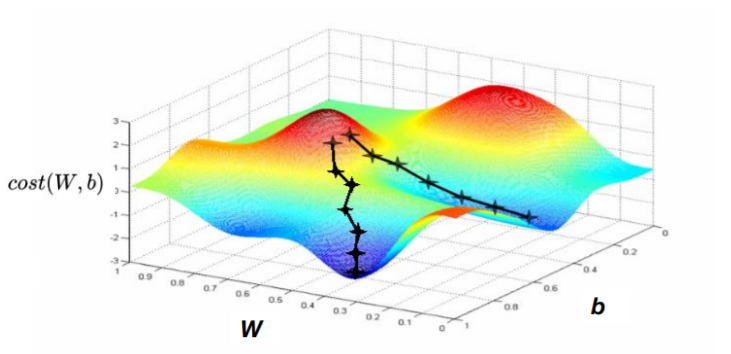
최초 추정 w,b값을 가짐. W,b값을 지속적으로 업데이트 해가면서 최소화 되는 값으로 업데이트 ->꼭지점을 찾을때까지 반복

w값에 기울기를 곱해서 w값을 빼준다. = 이 w가 다음 w값이 된다.-> 기울기가 0이 되는 지점을 찾는다.=>이는 비용이 최소화되는 지점이다.(기울기가 양수일경우. 기울기가 음수면 w값을 더해준다)

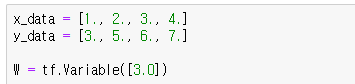
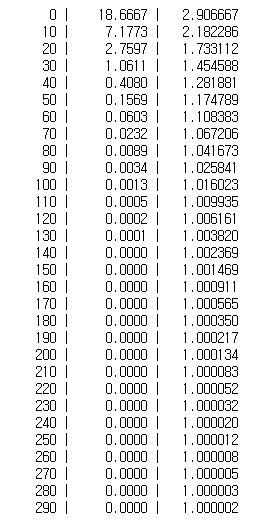
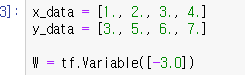
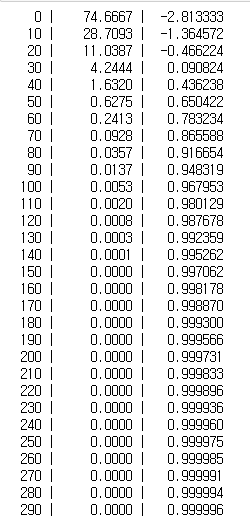
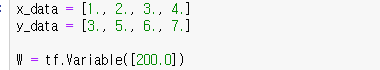
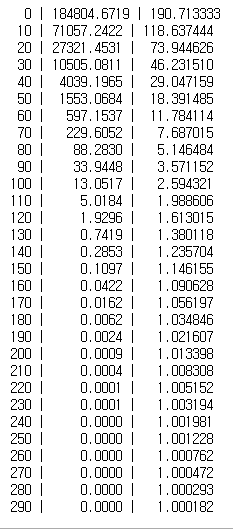
기울기(gradient)는 미분을 이용하여 구한다. Cost(w)를 w로 미분한 값.

 요 수식이 gradient descent를 표현하는 식(알파는 learning rate = 학습률)

기울기가 0이되는 지점이 여러 개가 되는데 이는 local minimam. 그 중 가장 낮은 점을 global minimam.



이는 local minimam과 global minimam이 일치하는 경우.

step | cost | W

w값에 상관없이 같은 값으로 수렴하나 w값이 커질수록 수렴하는 시간이 오래걸린다.