Pandas

1. 데이터 읽어온다. (pd.read\_csv(‘~~~.csv’)

Series는 컬럼하나 의미

검색 알고리즘

* 검색 -> 정렬(메모리 요구. 적게 쓰는게 좋음)

#이진 검색(가장 대표적, 실무에서 많이 씀)

이진 검색트리는 정렬동반하지 않음. N.P 문제가 생김

N.P 를 제한하면 풀림

지역 최적화를 시킴.

#코딩 테스트 관련

1. 그래프 문제 -> 트리로 해결
2. 트리 문제 -> 트리로 해결(자바는 재귀(x))
3. 정렬 관련문제는 x -> 검색문제
4. Linkedlist, stack, queue 만들어 쓰다가 망함.(기본 자료구조는 반드시 언어에서 제공하는 걸 사용)

Db에서 쿼리문이 검색

**힐클라이밍 알고리즘**

최대 최소 찾기. **딥러닝에 쓰임.**

Gradient boost. 확률적 경사 하강법

딥러닝은 위에서 아래로 하강.

오차가 가장 적은 최저점 찾는다.

외판원 문제 – 계산 복잡도 이론

N 퀸 문제

pip install pytest

테스트 주도 개발

For / while –> 일반 반복문

Foreach … -> 반복자. –generator로 만들 수 있다

Async / await - db에서 보여줄 양이 너무 많을 때

Exec – 문장이 들어감(print 등)

Eval – 순수함수(값음 반환)

Exec(print(“sds”))

Print(eval(“sadsad”))

!!!함수 연결해서 사용하는 방법 숙지 function call

추상화 – 어떤 정보를 표시하고 숨길지 결정

헙력 – 요청과 응답을 통해 객체끼리 협력. 객체는 각자 고유한 역할 담당

객체지향은 객체를 지향하는 것이지 클래스를 지향하는 게 아님

함수를 작게 만든다

#언덕등반 알고리즘

특정 좌표(x, y)를 기준으로 가장 낮/높은 값을 찾는다

좌표를 기반으로 좌우 값을 좀 더 살펴본다

그래프

간선과 정점

방향성 있는 것 - 네트워크

방향성 없는 것 – 순수 그래프(조합 그래프)

Tsp 문제는 간선의 합이 최소임을 원함

50개는 가능. 일반화된 상태에서는 불가능.

N vs n.p 문제임

시작지점 -> 랜덤 사용

랜덤을 사용해서 그나마 값이 적정값이라 가정

랜덤으로 하는데, 좌우를 둘러본다

우리는 지역 최적을 검색하려는데 전역 최적 알고리즘을 가져옴

다시 정리

Tsp

#언덕 등반

1) 정점을 구하는데 -> 인접행렬 구해야함(G->M, M->G, NetworkX)

2) 거리 -> 맨하탄, 유클리드 2 종류가 있다. -> 인접행렬과 같은 형태의 테이블로 만들어야함

3) 계산(최단거리) -> 최적인지 판단

First Choice

언덕등반 그대로 가져온다 ( 전역 검색)

거기다가 + mutaion(임의의 자료, noise)

유전 알고리즘?

Networkx

**[:, :] literally means [all rows, all columns]. – ‘,’ 기준으로 행과 열 구분**

[:, -1] it means you are taking all the rows and only the last column. -1 represents the last column.

When you do [:, :-1], it means you are taking all the rows and all the columns except the last column.

**Axis = 0 : 열 기준**

**Axis = 1 : 행 기준**

**0420 – ai 프로그래밍**

**언덕등반 알고리즘**

**Local search**

**선의 최고점 or 최저점 탐색**

**시작 지점에 따라 최적점(optimum) 못찾는다**

**경사하강법**

**미분 – 기울기 구함**

**하강 – 손실함수 최소화**

#Matplotlib  
# - subplots(여러개)  
# - subplot2grid

**힐클라임 쓰면 지역 최적은 찾을 수 있다.**

**구간 별로 전역 최적을 찾아야한다**

**전역을 탐색하고**

**공식문서를 봐라**

**유전 알고리즘 – 임의로 어떤 값을 맘껏 변경해보면 더 좋지 않을까?**

**프로그램만들 때 뭐부터 만드니?**

**Case1 ) 상태를 결정 -> c/c++**

**Case2 ) 행위를 결정 -> java, c#**

**Case x) 다 틀렸고 모든건 순수해야한다. -> react**

**사용자가 내 프로그램을 어떻게 사용할지 생각해야 함**

**경사하강법은 지역검색.**

**빠르지만 최적해는 포기**

**최적해 구하려면 전역검색(ex) 유전알고리즘)**

**배치사이즈 작으면 오래걸림**

**크면 해를 못구할 수 있다**

**데이터 정의 – 데이터 전처리 – 모델 작성 및 평가 – 반응형 시각화 – REST API 제공**

**결제모듈 달아 봐야함 – 사업자 내야함**

**서울에 있는 중소를 가라**