기계학습

AI프로그래밍을 위한 기본용어

기계학습을 위한 기본 상식

- 중앙처리장치(CPU) 와 그래픽처리장치 (GPU)
- 오픈소스소프트웨어 (Open Source Software, OSS)
- 통합개발환경 (IDE)
- 파이참(PyCharm)
- 넘파이(Numpy)
- 싸이파이(Scipy)
- Matplotlib
- 판다스(Pandas)
- 싸이킷런(Scikit-learn, Sklearn)
- CUDA와 cuDNN
- 파이썬 표준 내장 라이브러리

CPU와 GPU

- CPU (Central Processing Unit)
  - CPU는 입출력장치, 기억장치, 연산장치를 비롯한 컴퓨터 리소스를 이용하는 중앙 처리 장치로써, 컴퓨터의 두뇌와 같은 역할을 담당함
  - 멀티태스킹을 위해 나눈 작업들의 우선순위를 지정하고, 가상 메모리를 관리하는 등 컴퓨터를 지휘하는 역할을 수행함
  - 컴퓨터 프로그램의 대부분은 복잡한 순서를 가진 알고리즘을 가지고 동작하므로 CPU가 적합함

- GPU (Graphics Processing Unit)
  - 비디오, 즉 픽셀로 이루어진 영상을 처리하는 용도로 탄생했음
  - GPU는 반복적이고 비슷한, 대량의 연산을 수행하며 이를 병렬적으로 나누어 작업하기 때문에 CPU에 비해 속도가 무척 빠름
  - 렌더링을 비롯한 그래픽 작업의 경우 픽셀 하나하나에 대해 연산을 하기 때문에 연산 능력이 비교적 떨어지는 CPU가 GPU로 데이터를 보내 재빠르게 처리함

CPU와 GPU

- **CPU와 GPU 차이점 비교**
  - CPU 는 **순차적인 작업 (Sequential task)** 에 더 강점이 있는 반면, GPU 는 **병렬적인 작업 (Paralell task)** 에 더 강점이 있음
    - 현재 PC 에서 사용되는 CPU 의 코어는 보통 4~10개 정도이며 hyperthreading 기술을 통해 thread 를 2배 정도 늘릴 수 있음. 예를 들어 8 코어 16 threads CPU 의 경우, **병렬적으로 16개의 task 를 수행**할 수 있다는 것을 의미함
    - NVIDIA 의 2080 TI 의 경우 4,352 개의 코어를 갖고 있어 4,352개의 task를 병렬적으로 수행할 수 있음. CPU의 threading 기술을 감안하더라도 CPU와 GPU 의 코어 수의 차이는 200 배 이상 임. (4352/16 = 272)

  - 어플리케이션의 연산집약적인 부분을 GPU로 넘기고 나머지 코드 만을 CPU에서 처리하는 **GPU 가속 컴퓨팅**은 **딥러닝, 머신러닝 영역에서 강력한 성능을 제공**함. 즉, 사용자 입장에서 연산 속도가 놀라울 정도로 빨라짐

오픈소스

- **오픈소스 소프트웨어(Open Source Software, OSS)란?**
  - 오픈 소스 소프트웨어는 무료로 사용 가능하기 때문에 프리웨어와 혼동하는 경우가 많지만, 프리웨어는 무료로 사용 가능한 프로그램이고, 오픈 소스는 소스가 공개된 프로그램이므로 서로 다른 개념임
  - 프리웨어는 대부분 상업적 이용이 불가능하지만 오픈소스는 상업적으로 많이 활용됨
  - 일반 사용자 입장에서는 프리웨어나 오픈소스 소프트웨어나 비슷할 수 있으나, 소스 코드를 보고 이해할 수 있고 수정할 수 있는 개발자 입장에서는 크게 다름
  - SW 산업 초기에는 SW 개발 기업들이 소스코드를 공개하지 않고 철저 하게 보호하였으나 해커와 프로그래머들로 이루어진 그룹들이 소프트웨어는 무료로 배포되어야 한다고 주장함 (대표적인 인물: 리처드 스톨만 & 리눅스 토발즈)

"소프트웨어를 만드느라 애쓴 사람들에게 보상을 주지 않으면 도둑질이다."

- 빌 게이츠 1976년 취미생활자들에게 쓰는 공개 편지-

"소프트웨어는 누군가가 권리를 독점한 자산이 아니라, 공공으로 축적된 지식이자 공공재로서의 성격을 갖고 있다."

- 리처드 스톨만-

오픈소스

- **오픈소스의 성공 사례**
  - 리눅스의 성공은 오픈소스의 협업 모델이 상업적 기업보다 뛰어날 수 있다는 것을 증명하였고, 이후 많은 소프트웨어들이 오픈소스로 개발되어 오픈소스 시대를 열었음
  - 오픈소스는 빅데이터, 사물인터넷, 클라우드 등 IT 트렌드의 대세로 자리잡고 있으며, 안드로이드 뿐만 아니라 Axzure(클라우드), MySQL(데이터베이스), Apache(웹 서버), Hadoop(빅데이터) 등 거의 모든 분야에서 오픈소스 제품들이 큰 성과를 거두고 있음.

- **오픈소스 성공 이유?**
  - 성공적인 오픈소스 프로젝트는 체계적인 의사결정 시스템을 갖추고 있음
  - 분산버전컨트롤 시스템을 통한 "한계생산체감의 법칙" 극복
    - 한계생산체감의 법칙이란 사람이 많으면 많을 수록 그에 따른 부수적인 비용이 들어가기 때문에 생산성이 떨어진 다는 것으로 오픈소스프로젝트에 수많은 개발자들이 모일 경우 생산성이 떨어질 수 밖에 없다는 것
    - 토발즈는 기존에 있던 중앙 집중식 버전 관리시스템 (VCS)와 다른 분산 버전 관리 시스템(DVCS)인 Git을 개발함

오픈소스

- **오픈소스 성공 이유?**
  - **오픈소스의 경제학**
    - 프로그래머들도 오픈소스프로젝트에 참여함으로써 물질적인 보상을 얻기도 하고, 오픈소스 커뮤니티들도 수익을 창출할 수 있음
      - 오픈소스 기여는 개발자들에게 경험과 포트폴리오가 됨
    - 기업용 오픈소스 시장을 통해 돈을 벌 수 있음.
      - 대표적인 비즈니스 모델은 **교육**이며, 관련된 책, 강좌, 전문가 인증 제도 등을 통해 수익 창출이 가능함.
      - 두번째 모델은 **기술지원**이며, 기업용 오픈소스소프트웨어의 경우 업데이트 주기가 빨라 다른 시스템과의 호환에 문제가 발생할 수 있음. 이런 문제를 유지보수 해주며 수익 창출이 가능함.

AI프로그램을 위한 통합개발환경

- **통합개발환경의 이란?**
  - 개발자는 통합개발환경(IDE)를 이용해 프로그램을 개발함
  - 일반 편집기(메모자, 에디터)와 달리 변수나 함수 이름을 자동으로 완성하는 등 의 추가적인 기능을 제공하고 있어 개발자가 더욱 빠르게 코드를 작성할 수 있음
  - 코드의 실행과 결과 확인을 통합개발환경 내에서 한번에 처리할 수 있음
  - 프로그램이 정상적으로 동작하지 않을 때 디버거를 이용해 코드를 쉽게 분석할 수 있는 기능도 제공함
  - 파이썬용 IDE에는 주피터 노트북, 비주얼 스튜디오 기반의 PTVS, 이클립스 기반의 PyDev, 그리고 가장 많이 사용되는 파이참(PyCharm) 등이 있음

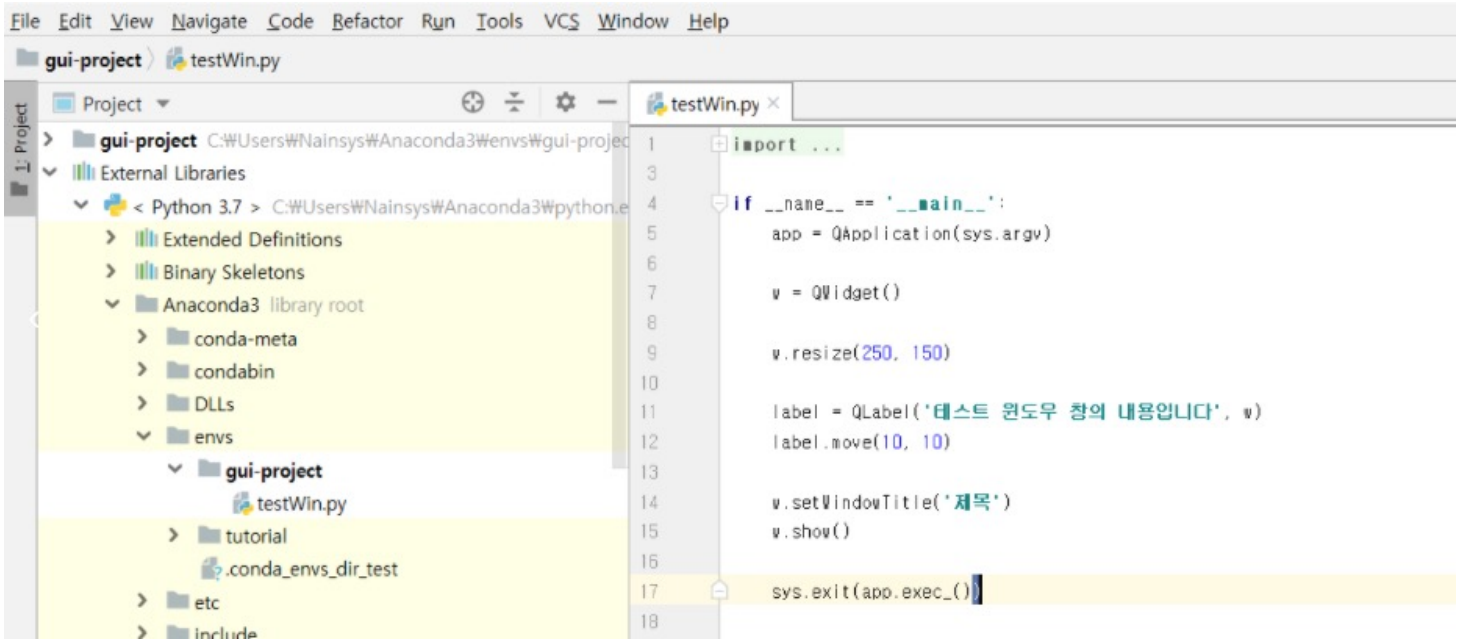
PTVS: http://microsoft.github.io/PTVS/

PyDev: http://www.pydev.org/

PyCharm: https://www.jetbrains.com/pycharm/download/

파이참

- **파이참(PyCharm) 이란?**
  - 파이참은 파이썬 프로그램을 쉽게 개발할 수 있도록 지원 하는 통합개발환경(IDE)임
  - 코드 편집기, 디버거, 컴파일러, 인터프리터 등을 포함하여 개발자에게 제공함
  - 현재 파이썬 개발 툴 중에서 높은 완성도를 지니고 있어 많이 사용되는 개발 툴임
  - PyCharm은 커뮤니티 에디션(무료)과 프로페셔널 에디션(유료)로 나뉘며, 윈도우/리눅스/맥 모두를 지원함



넘파이

- **넘파이(NumPy) 란?**
  - 파이썬으로 수치해석, 통계 관련 과학 계산을 하려면 필요한 패키지
  - 행렬이나 대규모 다차원 배열을 쉽게 처리할 수 있도록 지원하는 라이브러리
  - C, C++, FORTRAN 등의 언어에 비하면 편리하게 수치해석을 실행 할 수 있음
  - 더욱이 Numpy 내부 상당부분은 C나 FORTRAN으로 작성되어 있어 실행 속도가 빠름
  - Scipy, Pandas, matplotlib 등 다른 파이썬 패키지와 함께 쓰이는 경우가 많음

싸이파이

- **싸이파이(SciPy) 란?**
  - Scipy는 수치해석을 위한 과학 계산용 함수를 모아놓은 파이썬 패키지
  - Scipy는 고성능 선형대수, 함수 최적화, 신호 처리, 특수한 수학 함수와 통계 분포 등을 포함한 많은 기능을 제공함
  - Scikit-learn은 알고리즘을 구현할 때 SciPy의 여러 함수를 사용함
    - 예) scipy.sparse 를 통해 데이터를 표현하는 방법인 희소 행렬 기능을 제공
  - Numpy와 Scipy는 서로 떨어질 수 없을 정도로 밀접한 관계가 있으며 Scipy를 활용할 때에는 상당히 많이 Numpy를 이용하게 됨
  - Scipy를 이용하면 Numpy만으로 길게 코딩해야 하는 기법들을 단 2~3줄로 구현 가능

Matplotlib

- **Matplotlib 이란?**
  - Matplotlib은 파이썬의 대표적인 과학 계산용 그래프 라이브러리
  - 선 그래프, 히스토그램, 산점도 등을 지원하며 출판에 사용될 수 있을 만큼의 고품질 그래프를 그려줌
  - 파이썬에 기본적으로 내장된 리스트형 자료로도 충분히 수많은 종류의 데이터를 그래프화 할 수 있으나 Numpy를 쓸 때보다는 코드가 복잡해지고, 그래프를 얻는 속도도 느림.
  - Matplotlib 사용 시 Numpy의 ndarray 자료형을 사용하고 리스트형은 사용하지 말 것
  - 데이터와 분석 결과를 다양한 관점에서 시각화해보면 매우 중요한 결과와 통찰을 얻을 수 있음

Pandas

- **Pandas 란?**
  - 판다스란 데이터 처리와 분석을 위한 파이썬 라이브러리로, 행과 열로 이루어진 데이터 객체를 만들어 다룰 수 있어서 안정적으로 대용량의 데이터들을 처리하는데 매우 편리한 도구임
  - 우리가 가장 많이 접하는 데이터는 행과 열로 되어 있는 엑셀의 스프레드시트 형태임
  - 판다스는 크게 1차원 자료구조인 Series, 2차원 자료구조인 DataFrame, 그리고 3차원 자료구조인 Panel 을 지원함
  - Pandas를 사용하면 엑셀 쉬트 같은 2차원 테이블 데이터를 파이썬에서 편리하게 사용할 수 있음

Scikit-learn

- **Scikit-learn 이란?**
  - 싸이킷런은 2007년 구글 썸머 코드에서 처음 구현되었으며 현재 파이썬으로 구현된 가장 유명한 기계학습 오픈소스 라이브러리임
  - 싸이킷런의 장점은 라이브러리 외적으로는 scikit 스택을 사용하고 있기 때문에 다른 라이브러리와의 호환성이 좋음
  - 내적으로는 통일된 인터페이스를 가지고 있기 때문에 매우 간단하게 여러 기법을 적용할 수 있어 쉽고 빠르게 결과를 얻을 수 있음
  - 싸이킷런은 파이썬 머신러닝 라이브러리에서 정석과도 같은 라이브러리임
  - 다양한 분류기를 지원하며 머신러닝 결과를 검증하는 기능도 가지고 있음
  - 분류, 회기, 클러스터링, 차원 축소 처럼 머신러닝에 자주 사용되는 다양한 알고리즘을 지원함

CUDA & cuDNN

- **CUDA 와 cuDNN 이란?**
  - CUDA는 그래픽 처리 장치에서 수행하는 알고리즘을 C 프로그램밍 언어를 비롯한 산업 표준 언어를 사용하여 작성할 수 있는 GPGPU 기술 임
  - CUDA는 엔비디아가 개발해오고 있으며, 이 아키텍처를 사용하려면 NVIDIA GPU와 특별한 스트림 처리 드라이버가 필요함
  - CUDA 플랫폼은 컴퓨터 커널의 실행을 위해 GPU의 가상 명령 집합과 병렬 연산 요소들을 직접 접근할 수 있는 소프트웨어 계층임
  - 딥러닝을 도와주는 여러 라이브러리도 CUDA와 함께 제공됨.
  - cuDNN, Convolution 연산을 더 빠르게 만들어 주는 cuFFT, 선형대수 모듈인 cuBLAS 등 사실상 필요한 라이브러리들은 대부분 구현되어 있음
  - NVIDIA 이외의 그래픽카드에서 작동하지 않음
  - NVIDIA 이외의 그래픽카드에서 병렬연산을 하고 싶으면 OpenCL을 사용하면 됨
  - 애플, 인텔, AMD, ARM 등에서는 엔비디아를 견제하기 위해 OpenCL을 밀고 있음

파이썬 표준 내장 라이브러리

- **파이썬 표준 내장 라이브러리**
  - 파이썬의 표준 라이브러리는 매우 광범위하며 다양한 기능을 제공함
  - 내장이란 파이썬이 기본적으로 제공하는 것으로 import를 이용해 로드하지 않아도 사용할 수 있는 것들을 말함
  - 이 API들은 응용 프로그램이 시스템에 접근할 수 있는 필수적인 기능을 제공함
  - 객체를 문자로 출력하거나, 절대값을 구하거나, 파일을 제어할 수 있는 함수와 같이 자주 사용되는 함수들이 여기에 속함
  - 파이썬 표준 라이브러리는 다음을 참고하면 됨
    - https://docs.python.org/ko/3/library/index.html