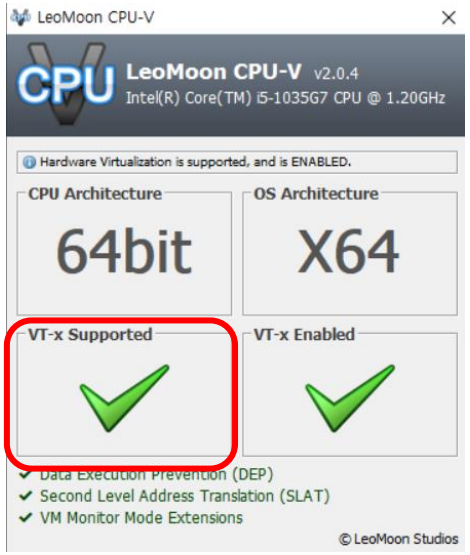
머신러닝 기본개념 소개

팔복기술

2021

1. 코딩환경 구축 – WSL2 설치 - **설치 환경 검사**

- 실습을 위해 WSL2(Windows Subsystem for Linux 2) 설치가 가능해야 함.
- Windows 상에서 Linux를 사용할 수 있도록 해주는 유틸.
- Machine Learning 라이브러리와 정합성이 좋고 수행속도가 빠름.



- WSL2 설치가능여부 확인
  - ✓ https://leomoon.com/downloads/desktop-apps/leomoon-cpu-v/ → LeoMoon CPU-V.exe 다운로드 → 실행
  - ✓ 그림과 같이 VT-x Supported 체크되어 있어야 함.

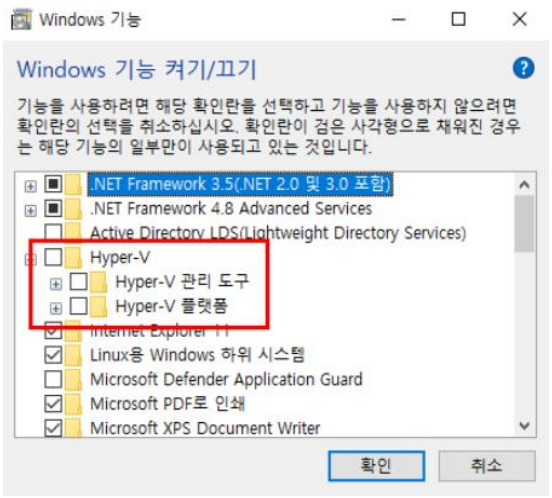
- VT-x Supported 체크되어있지 않으면 wsl 설치가 불가능한 컴퓨터임 !!!
- 또한 wsl 과 윈도우가 파일시스템을 공유하기 위해서는 윈도우 파일시스템은 NTFS(Windows 10 에서는 기본설정) 여야 함.

- VT-x Enabled 설정
  - ✓ 명령창에서 $ systeminfo
  - ✓ 내용 중 Hyper-V 요구사항 → 펌웨어에 가상화 사용 : '예'/'아니오' 확인
  - ✓ '아니오'인 경우 부팅시 BIOS 들어가서(Del, F2, F10 등) VTx, Virtual Technology 등으로 표시된 항목 Enabled 로 설정
  - ✓ 참고 : ASUS Bios의 경우 설정 메뉴
  - ✓ Advanced Mode > Advanced > CPU Configuration > Intel Virtualization Technology : Enabled

1. 코딩환경 구축 – WSL2 설치 - **WSL 설치**

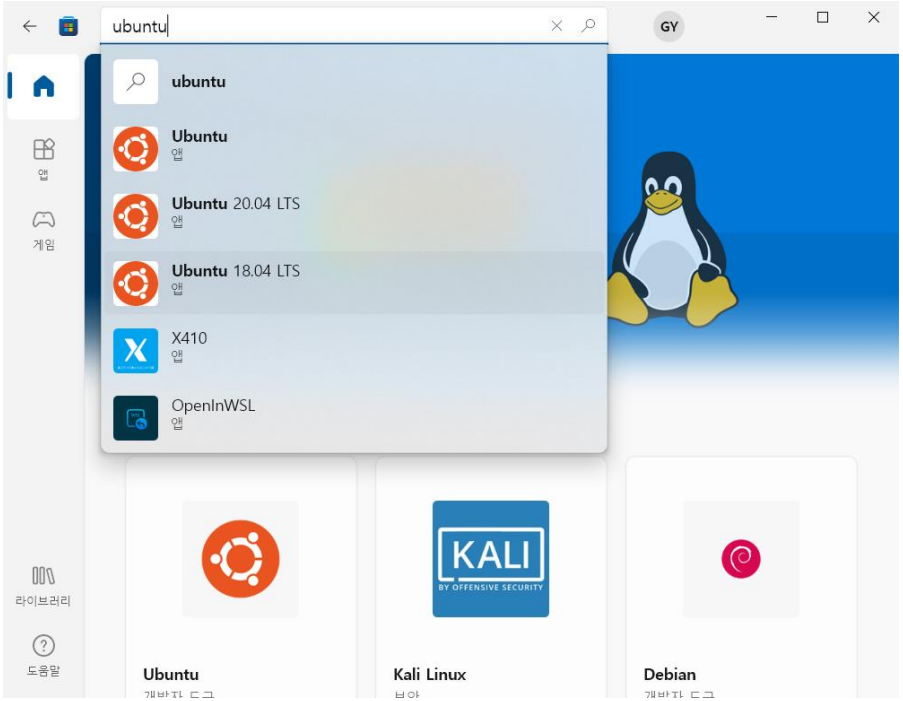
- 윈도우 업데이트
  - ✓ window key + R → winver
  - ✓ 버전 2004 이상, 빌드 19041 이상이어야 함.
  - ✓ 업데이트 시 Windows 11 을 설치하지 않도록 주의
  - ✓ 수동업데이트 링크 : https://www.microsoft.com/en-us/software-download/windows10

- WSL2 설치
  - ✓ powershell 관리자모드 에서
  - ✓ $ dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart
  - ✓ $ dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart
  - ✓ Reboot
  - ✓ 검색 : “Windows 기능 켜기/끄기” 또는 “optionalfeatures”
  - ✓ 체크 해제 : Hyper-V
  - ✓ 체크 : Linux용 Windows 하위시스템, 가상머신 플랫폼
  - ✓ download WSL2 Linux kernel update package for x64 machines -> 설치
  - ✓ (파워쉘) $ wsl --set-default-version 2



1. 코딩환경 구축 – WSL2 설치 - **WSL 설치**

- https://aka.ms/wslstore
- 여기서 Ubuntu 선택하면 최신버전 설치됨 →
- Ubuntu 검색
- Ubuntu 20.04 LTS 선택 → 다운로드 → 열기



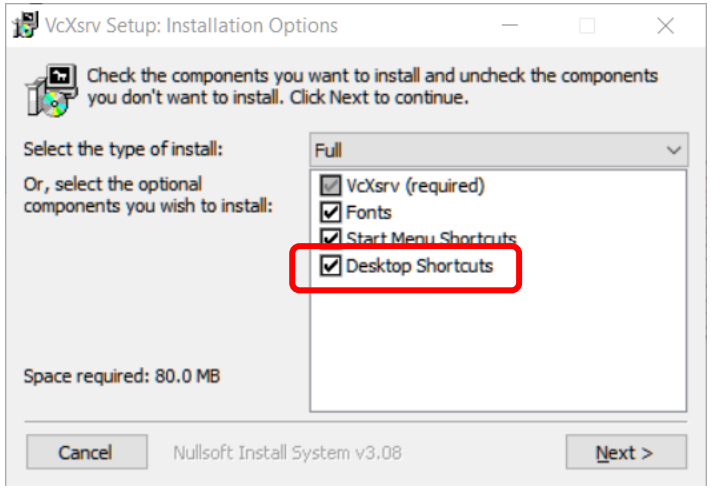
- Enter new UNIX username : <원하는 이름>
- Enter new UNIX password: <원하는 비밀번호>
- pi@DKT-JJ:~$
- → '$' 나오면 설치 완료된 것임.

1. 코딩환경 구축 – Windows terminal 설치

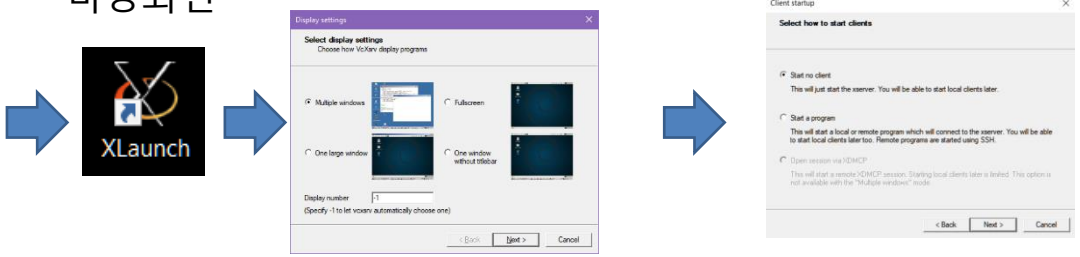
- 터미널 설치
- https://aka.ms/store
  - ✓ Store 에서 Windows Terminal 검색 → 다운로드 → 설치
  - ✓ 빠른 실행 또는 바탕화면에 Windows terminal 추가
  - ✓ (파워쉘) wsl –l –v # 실행버전 확인.
- 패키지 현행화
  - ✓ (wsl) $ sudo apt update && sudo apt upgrade

1. 코딩환경 구축 – gui 환경구축(디스플레이서버 설치)

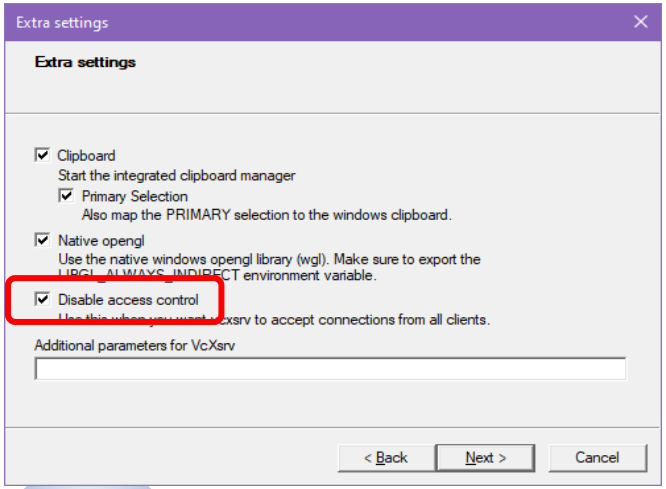
✓ https://sourceforge.net/projects/vcxsrv/ → vcxsrv-64.1.20.14.0.installer.exe



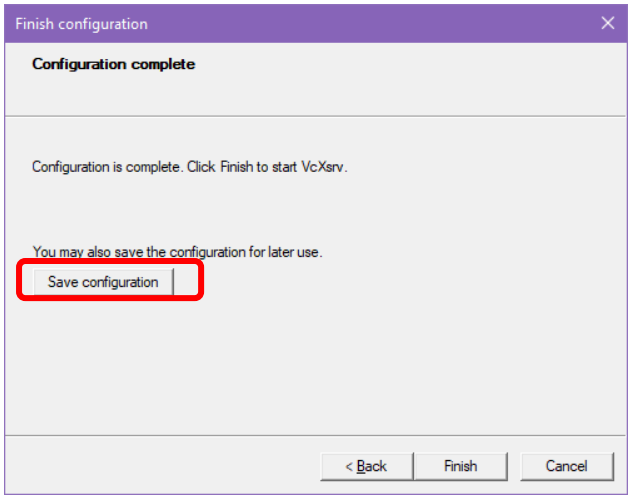
바탕화면



Check Disable access control

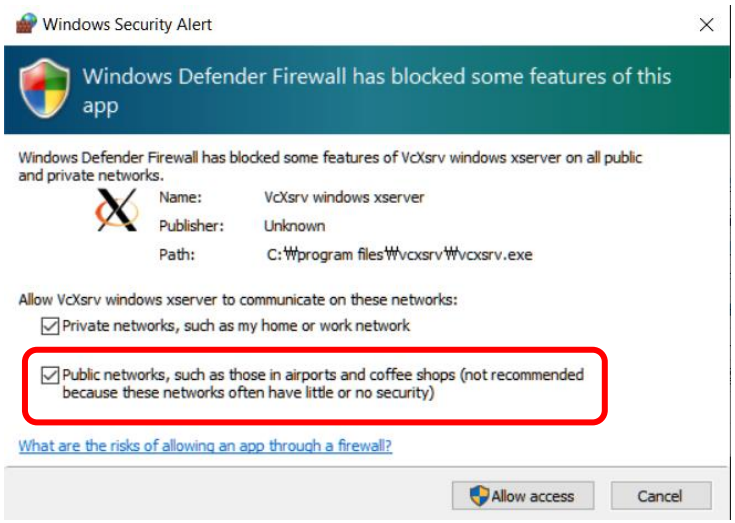


윈도우 시작시 자동실행을 위해 저장



1. 코딩환경 구축 – gui 환경구축

Public inbound 통신 허용



- 윈도우 시작시 VcXsrv 자동실행
  - ✓ Window + R > shell:startup
    - → open C:\Users\{User}\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup
  - ✓ 종료시 저장한 Config.xlaunch 파일을 Startup 폴더로 옮김.

1. 코딩환경 구축 – gui 환경구축

wsl 에서 설정

- $ sudo apt install kate # 문서 편집기 설치
- $ explorer.exe .
- .bashrc 파일에 아래 내용 추가
- export DISPLAY=$(cat /etc/resolv.conf | grep nameserver | awk '{print $2}'):0

  참고) export DISPLAY=$(awk '/nameserver / {print $2; exit}' /etc/resolv.conf 2>/dev/null):0

  export LIBGL_ALWAYS_INDIRECT=1
- $ source .bashrc
- < test >
- $ sudo apt install x11-apps -y
- $ xeyes
- Cf. $ wf.msc → Inbound Rules → VcXsrv / Public > Allow the Connection

1. 코딩환경 구축 – gui 환경구축(크롬 설치)

```
#install packages
$ sudo apt-get install -y curl unzip xvfb libxi6 libgconf-2-4 fonts-liberation
#get latest chrome
$ wget https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb
#install it
$ sudo apt install ./google-chrome-stable_current_amd64.deb

$ google-chrome
```

1. 코딩환경 구축 – Anaconda 소개

- Anaconda 소개
- Python은 수치 및 과학 응용 프로그램에 주로 사용됨.
- 효율적인 방식으로 수치 계산을 수행하기 위해 Python은 포트란 언어를 사용하여 부분적으로 구현되는 NumPy 라이브러리 와 같은 다른 언어로 구현되는 외부 라이브러리에 의존함
- 이러한 종속성으로 인해 때때로 필요한 모든 라이브러리를 연결하여 수치 계산을 위한 환경을 설정하는 것이 쉽지 않음.
- Anaconda는 이러한 종속성을 숨기고 단일 인터페이스를 통해 관리할 수 있도록 해줌.

1. 코딩환경 구축 – Miniconda 소개

- Conda
  - 아나콘다 또는 Miniconda 유통없이 설치 될 수있는 패키지 의존성 및 환경 관리 시스템.
  - Python 자체와 여러 타사 오픈 소스 프로젝트용 바이너리를 포함하여 PyData 생태계에서 소프트웨어의 전체 배포
  - 주요 목적은 사전 컴파일된 소프트웨어 버전을 다운로드하여 외부 종속성 문제를 쉽게 해결함.
- Miniconda 설치
  - Anaconda는 사전 설치된 많은 패키지가 있는 환경을 제공하며 그 중 많은 패키지가 사용되지 않음.
  - Miniconda는 미니멀하고 깨끗하며 Anaconda의 모든 패키지를 쉽게 설치할 수 있음.
  - https://docs.conda.io/en/latest/miniconda.html
  - Linux > Python 3.8 > Miniconda3 Linux 64-bit 선택(tensorflow, Pycaret, autokeras 호환성 유지)
  - $ wget https://repo.anaconda.com/miniconda/ Miniconda3-py38_4.10.3-Linux-x86_64.sh
  - cf. Py38 설치안되면 latest 설치 $ wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_(

Linux

Python version	Name	Size	SHA256 hash
Python 3.9	Miniconda3 Linux 64-bit	63.6 MiB	1ea2f885b4dbc3098662845560bc64271eb17085387a70c2ba3f29fff6f8d52f
	Miniconda3 Linux-aarch64 64-bit	62.6 MiB	4879820a10718743f945d88ef142c3a4b30dfc8e448d1ca08e019586374b773f
	Miniconda3 Linux-ppc64le 64-bit	60.6 MiB	fa92ee4773611f58ed9333f977d32bbb64769292f605d518732183be1f3321fa
	Miniconda3 Linux-s390x 64-bit	57.1 MiB	1faed9abecf4a4ddd4e0d8891fc2cdaa3394c51e877af14ad6b9d4aadb4e90d8
Python 3.8	Miniconda3 Linux 64-bit	98.8 MiB	935d72deb16e42739d69644977290395561b7a6db059b316958d97939e9bdf3d
	Miniconda3 Linux-aarch64 64-bit	94.8 MiB	19584b4fb5c0656e0cf9de72aaa0b0a7991fbd6f1254d12e2119048c9a47e5cc
	Miniconda3 Linux-ppc64le 64-bit	93.3 MiB	c1ac79540cb77b2e0ca5b9f78b3bc367567d810118500a167dea4a0bcab5d063
	Miniconda3 Linux-s390x 64-bit	89.0 MiB	55f514110a50e98549a68912cbb03e43a36193940a1889e1c8beb30009b4da19
Python 3.7	Miniconda3 Linux 64-bit	84.9 MiB	a1a7285dea0edc430b2bc7951d89bb30a2a1b32026d2a7b02aacaaa95cf69c7c
	Miniconda3 Linux-aarch64 64-bit	89.2 MiB	65f400a906e3132ddbba35a38d619478be77d32210a2acab05133d92ba08f111
	Miniconda3 Linux-ppc64le 64-bit	88.1 MiB	e4f8b4a5eb8da1badf0b0c91fd7ee25e39120d4d77443e7a1ef3661fd439a997
	Miniconda3 Linux-s390x 64-bit	84.1 MiB	7ab9f813dd84cb0951a2d755cd84708263ce4e03c656e65e2fa79ed0f024f0f7

1. 코딩환경 구축 - Miniconda 설치

- # ~/Downloads/ Miniconda3-py38_4.10.3-Linux-x86_64 .sh 로 저장됨.
- # (만약 없으면 링크를 마우스 우클릭후 save link as로 저장)
- $ bash Miniconda3-py38_4.10.3-Linux-x86_64 .sh
- (물어보는 프롬프트 : yes, enter…)
- $ source ~/.bashrc
- $ conda install pip ipykernel –y
- $ conda install nodejs -y # go_to_definition에 필요함. → conda update nodejs -y (최신 버전으로)

• Jupyter lab 설정

- $ conda install jupyterlab -y # base에 설치.
- #$ jupyter-notebook --generate-config
- #edit $HOME/.jupyter/jupyter_notebook_config.py
- #C.NotebookApp.notebook_dir = '/home/pi/study/Code' # # 없앨 것
- $ jupyter lab --no-browser → 127.0.0.1:8888/lab 결과 확인 필요.

- 노트북 사용법
- https://www.youtube.com/watch?v=70sRgL42c1w
- https://www.youtube.com/watch?v=UnXXH72-ENc
- 사용법 단축키 소개(9. jupyterlab.jpg)

1. 코딩환경 구축 – 가상환경 생성

- Code completion 설치(base 환경에서)
  - $ pip install 'jupyterlab>=3.0.0,<4.0.0a0' jupyterlab-lsp
  - $ pip install 'python-lsp-server[all]'
  - $ jupyter labextension install @krassowski/jupyterlab_go_to_definition
  - # ipython profile create
  - Edit /home/pi/.ipython/profile_default/ipython_config.py
  - edit c.IPCompleter.use_jedi = False
  - Restart JupyterLab
- jupytext 설치 : pip install jupytext
- 가상환경 생성
  - $ conda config --add channels conda-forge → conda config --show channels # default channel 변경
  - $ conda create –-name study python=3.8 pip ipykernel # -- 다시 확인할 것.
  - $ conda activate study
  - $ conda install pip ipykernel
  - $ python -m ipykernel install --user --name study # no module → pip install ipykernel
  - Check installed folder
- 필요 기능, 라이브러리 설치
  - $ conda install pip -y
  - $ pip install numpy scipy scikit-learn matplotlib ipython pandas pillow imageio
  - $ sudo apt-get install -y graphviz # pip 설치시 system path 설정해주어야 함.

1. 코딩환경 구축 – VSCode 설치

Windows VSCode 설치
  - 설치 중 추가작업선택 > add to path체크 확인.
윈도우 재시작

  확장설치
- Remote Deveplopment
      (includes - Remote wsl, Remote-SSH (for RPI), Remote Container (for docker inside))

< Open VSCode in wsl env>
  - cd ~/Code → code .
< Open VSCode in windows>
1. - Start VS Code.
2. - F1 > select **Remote-WSL: New Window** (default wsldistro)
3. - 파일 > 폴더열기 > /home/pi/Code

 - Python (from MS)
- Python Extension Pack > Python
  - Jupyter(MS)
  - Brachet Pair Colorizer
  - Rainbow CSV
  - 커널 선택(오른쪽 위) > study-tf

1. 코딩환경 구축 – 단축키 설정방법

F5 단축키 설정 (디버깅 실행, continue 겸용 방법)

- F1 > Preferences : Open keyboard Shortcuts 또는 ctrl + k ctrl +s
- 'debug'로 검색,
- Debug:계속 더블클릭 > F5 > '3개의 기존 명령에...' 클릭 >
- '언제' 우클릭 > 식인 경우 변경 >
- '디버깅 시작' : !inDebugMode, 계속 : inDebugMode
- '단위실행(StepOVer)' : F8, 중복키 정의는 들어가서 '키바인딩 제거'
- '단위실행(StepOVer)' : F8, 중복키 정의는 들어가서 '키바인딩 제거'

< VSCode for Python manual >
https://code.visualstudio.com/docs/python/python-tutorial
https://code.visualstudio.com/docs/python/editing
https://code.visualstudio.com/docs/python/linting
https://code.visualstudio.com/docs/python/debugging
https://code.visualstudio.com/docs/python/testing
https://code.visualstudio.com/docs/python/settings-reference

https://code.visualstudio.com/docs/datascience/data-science-tutorial

1. 코딩환경 구축 - 파이썬 머신러닝 생태계를 구성하는 주요 패키지

- 머신러닝 패키지
  - scikit learn
- 배열/선형대수/통계 패키지
  - Numpy : 벡터/행렬 연산에 사용
  - scipy : 파이썬의 대표적인 통계 패키지/자연과학 분야에서 사용
- 데이터 핸들링
  - pandas : 데이터 분석에 사용
- 시각화
  - plotly : 코드가 간편하고 개인적취향으로 더 이쁘다
  - matplotlib : 대표적
  - seaborn : matplotlib을 기반으로 다양한 테마,차트 기능 추가/matplotlib에 의존적
- 대화형 파이썬 툴
  - jupyter : 데이터 분석을 순차적으로 실행하면서 확인 가능/코드를 분할해서 수행 가능
- 딥러닝 패키지
  - TenserFlow : keras보다 더 디테일한 조작 가능
  - Keras : tenserFlow 위에서 동작/쉬움
  - Pytorch : 파이썬을 언어로 사용하여 쉬움

주차별 계획

21	환경설정 ML intro(4) - wsl, jupyter, 붓꽃 예제
22	지도학습 1(4) - 선형모델, 과대/과소적합, SVM
23	지도학습2, 비지도학습(5) - Random Forest., Boosting - 비지도 학습, 데이터 전처리
24	모델개선 및 성능평가(5)

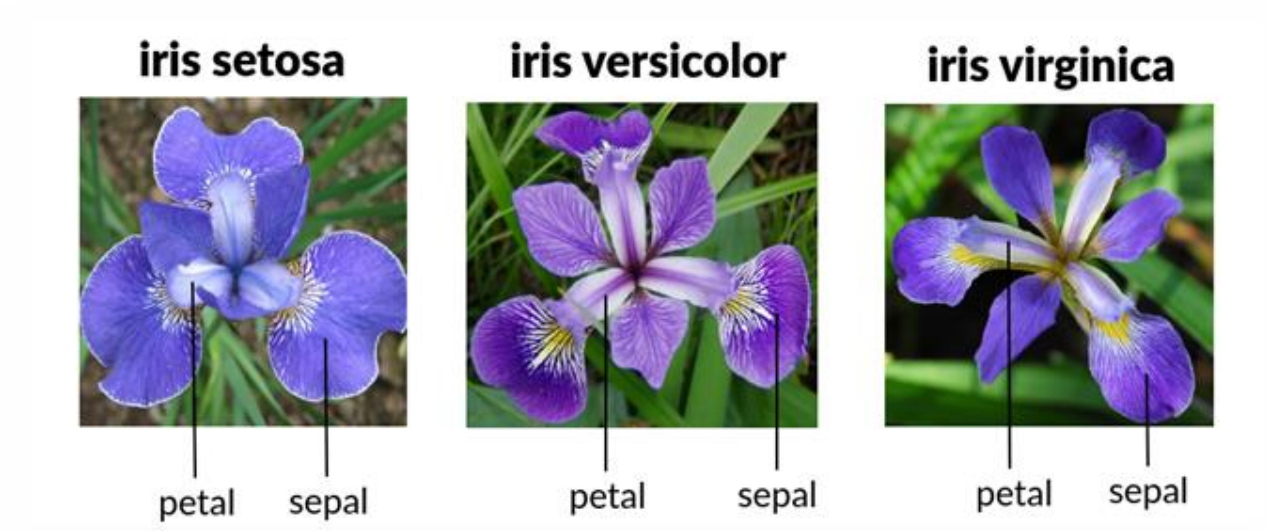
2. 머신러닝 간단예제

교재 저장소 : https://github.com/rickiepark/intro_ml_with_python_2nd_revised

- 머신러닝은 입력데이타를 기반으로 기대출력에 가깝게 만드는 유용한 표현을 학습하는 것.
- 머신러닝의 필요조건
  - 입력 데이타포인트(샘플) : 대상데이타 중 규칙을 찾기 위한 문제에 해당하는 데이터
  - 기대출력 : 입력데이타포인트와 짝이 되는 기대값
- 알고리즘 성능측정방법
  - 알고리즘의 현재출력과 기대출력(정답)간의 차이를 결정하는 방법
- 문제 유형
  - 분류(Classification) cf. 회귀(Regression)
- 레이블
  - 특정 데이터 포인트가 가질 수 있는 클래스값

2. 머신러닝 간단예제

- 문제
  - 붓꽃(Iris)의 품종 분류
  - 붓꽃 품종 종류 : versicolor, virginica, setosa
- 기초 용어
  - 클래스 : 분류 값(붓꽃 품종)
  - 문제 유형 : 분류(Classification) cf. 회귀(Regression)
  - 레이블 : 특정 데이터 포인트가 가질 수 있는 클래스값



2. 머신러닝 간단예제

< 3.0. First App.ipynb >

- 적용 머신러닝 모델 : 최근접 이웃 알고리즘
  - 단순히 훈련데이타를 저장하고 예측시 문제 포인트에서 가까운 포인트 개수를 세어 가장 많은 레이블을 예측값으로 함.
  - 학습가정은 없다 단순히 학습데이타를 저장하는 특징이 있음.

- 데이터셋은 두개의 넘파이배열로 이루어짐.
  - X : 학습데이타의 특성 정보를 담고 있는 데이터 샘플들의 집합.
  - Y : 각 특성샘플의 실제 목표값

- 훈련세트 : 훈련에 사용될 데이타셋.
- 테스트세트 : 생성된 모델이 새로운 데이터에 얼마나 잘 적용될 수 있는지 평가하기 위한 데이타셋

- 정확도 : 전체 학습/테스트 대상 샘플 중 정확하게 분류한 샘플의 비율

2. 코딩환경 구축 – matplotlib 에서 한글폰트 사용 설정

- 한글폰트 wsl에 설치

  → $ sudo apt install fonts-nanum

- Matplotlib 설정
  - $ sudo fc-cache –fv # 시스템 폰트캐쉬 삭제
  - $ rm –rf ~/.cache/matplotlib/* # 사용 폰트정보 삭제
  - mpl.fontmanager.findSystemFonts(fontpaths=None, fontext='ttf') # 시스템 폰트에 등록되었는지 확인
  - $ cd ~/miniconda3/envs/study/lib/python3.8/site-packages/matplotlib/mpl-data/ # matplotlib 폰트 등록
  - $ Explorer .
  - Matplotlibrc 파일 수정 (# 없애는 것 주의)
    - Font.family: NanumGothic
  - Jupyter lab reboot
  - import matplotlib as mpl # '-' 기호 깨지는 문제 해결
  - Mpl.rcParams['axes.Unicode_minus'] = False