리눅스로 한학기 살기

2016101950 유전공학과 이주영

Week1 – docker

데이터센터 프로그래밍 수업의 일환으로 docker를 깔아야 했다.

Window 10이지만 pro 버전이 아니었기 때문에 생각보다 너무 어려웠다.

교수님의 질의응답 사이트를 확인해보니 window에 까는 게 생각보다 복잡한 것 같았다.

그래서 VMware에 깔면 더 쉬울 거라 생각했다.

까는 도중에 “E: unable to locate package – “

오류가 반복적으로 발생하여 그 다음 과정을 진행할 수 없었다. 그래서 열심히 구글링을 하였다. 답답하다고 느꼈던 건 많은 글들이 현재 사이트와 다른 형태를 띄는 사이트에서 어떻게 하면 되는지를 설명하는 게 대부분이라는 것이었다.

시도를 하던 중 이 오류의 원인이 우분투 버전에 대한 패키지 지원 기간 만료로 package repository에 접근이 불가 한 것이기 때문이라는 글을 읽게 되었다. 이에 우분투 서버를 old-releases 서버로 변경하면 해결 될 것이라 하였다.

| code

\*서버변경을 위한 파일 오픈

$sudo cd /etc/apt/

$sudo vi sources.list

\*전환할 수 있는 code 사용해서 내용 수정

:%s/us.archive.ubuntu.com/old-release.ubuntu.com/g

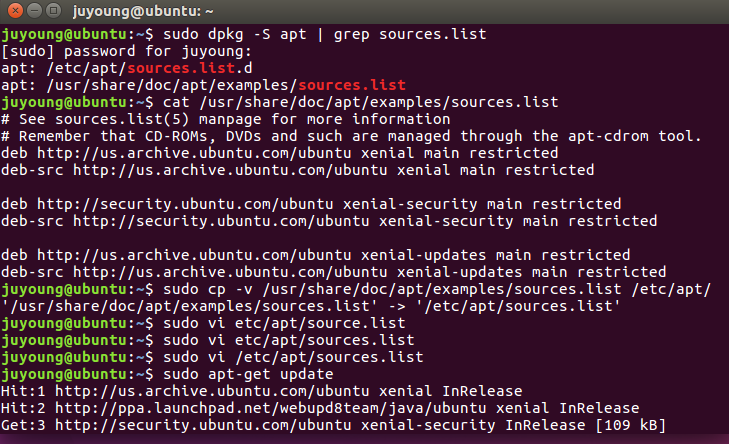
:%s/security.ubuntu.com/old-release.ubuntu.com/g

:%s/extras.ubuntu.com/old-release.ubuntu.com/g

\*update

$sudo  apt-get update

결과는 대 실패! 다시 생각해보니 깔고자 했던 게 구버전이 아니기 때문에 이 해결책은 소용이 없는 게 당연했다. 많은 사람들이 이걸로 해결됐다는 말에 혹해서 문제 원인 파악을 집중하지 못했던 것 같다. 이에 다시 수정을 하기위해 파일을 열었다가 아예 모든 내용을 old-release.ubuntu.com으로 바꿔버리는 실수를 하게 되었고 sudo apt-get update가 안되기 시작했다.



위 과정을 통해 해결!

나는 VMware를 리눅스와 동일시 하고 그렇게 깔면 될 거라고 생각했다. 여기까지 오류를 해결하는 과정에서 VMware에 docker를 까는 게 아니라 docker를 VMware에서 돌아가게 하는 것이라는 걸 인지하지 못했다. Virtual machine이라는 개념을 까먹고 있었던 것 같다. 그래서 결국 window에 tool box로 성공적으로 깔게 되었다.

@"%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile -ExecutionPolicy Bypass -Command "iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))" && SET "PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin"

choco install -y docker

choco install -y docker-machine

choco install -y docker-machine-vmwareworkstation

docker-machine --native-ssh create -d vmwareworkstation default

docker-machine env

추가적으로 위 과정을 통해 VMware에서 docker를 사용할 수 있게 되었다. 무서워서 여기서 별로 사용하고 싶지는 않다.

Week2 – FastQC

나는 bioinformatics를 전공하고 있다. 말그대로Bio와 informatics를 합친 분야로 인간의 생명과 관련한 방대한 데이터를 분석하여 더 나은 결과를 만드는 과정이다. 이번에 좋은 기회가 생겨 RNA-seq을 진행할 수 있게 되었다. 이 과정에서 raw data를 파싱하는데 이때 리눅스가 사용된다.

Ilumina sequencing을 진행하게 되면 raw data를 받게 된다.

이때 이 FastQC의 역할은 Raw data를 열어 얼마만큼의 정확도를 가지고 sequencing이 되었는지를 확인할 수 있게 해준다.

| code

sudo apt update

sudo apt install fastqc

sudo apt update

FastQC 사이트에 가서 확인해보니 설치법이 매우 간단하였다.

(새로운 유저를 만드는 법이 생각이 나지 않아 이걸 위해 새로 만들면서 진행)

sudo adduser -m 사용자 이름 [구버전]

sudo adduser 사용자 이름 [새로운 버전]

옵션 -m이 사라졌다.

그 다음 권한을 주기 위해 sudo visudo를 실행 시켜 아래와 같이 입력!



그 다음 sudo apt update를 한 후 fastqc를 깔려고 하는 순간 저번주에 보았던

“E: unable to locate package fastqc“ 를 또 보게 되었다…….

다시 sources.list를 열어서 아래를 추가로 입력을 하고 sudo apt update를 했는데 평소와 달리 너무 많은 시간이 걸렸다

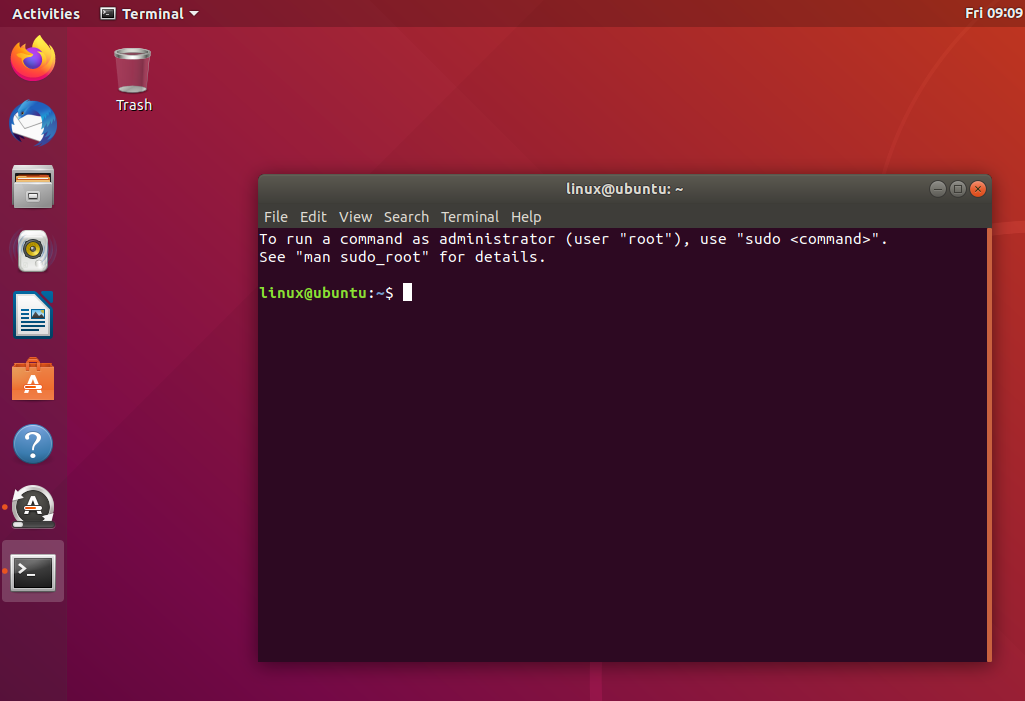
deb <http://archive.ubuntu.com/ubuntu> bionic main restricted universe multiversedeb <http://archive.ubuntu.com/ubuntu> bionic-security main restricted universe multiversedeb <http://archive.ubuntu.com/ubuntu> bionic-updates main restricted universe multiverse

이 후 fastqc 까는 건 성공적으로 이루어진 줄 알았는데 실행이 안된다.

이상하다 너무 느리다 sources.list를 열어 추가를 하는게 아니라 지우고 다시 써야 했던 것 같다. 찾아보니 이 sources.list란 apt-get으로 무언가를 불러올 때 가져오는 site를 말하는 것이고 간혹 한국어로 설정된 주소들을 kr->us로 바꿔주면 해결되는 것들을 보았을 때 이렇게 시간이 오래 걸리는 건 아마도 내가 주소를 여러 개로 둬서 여기도 갔다가 저기도 갔다가 해서 그런 게 아닐까 생각된다.

그런데 다 지우고 다시 했는데 그것도 안된다. 껐다 켰는데 터미널 창이 켜지지가 않는다.

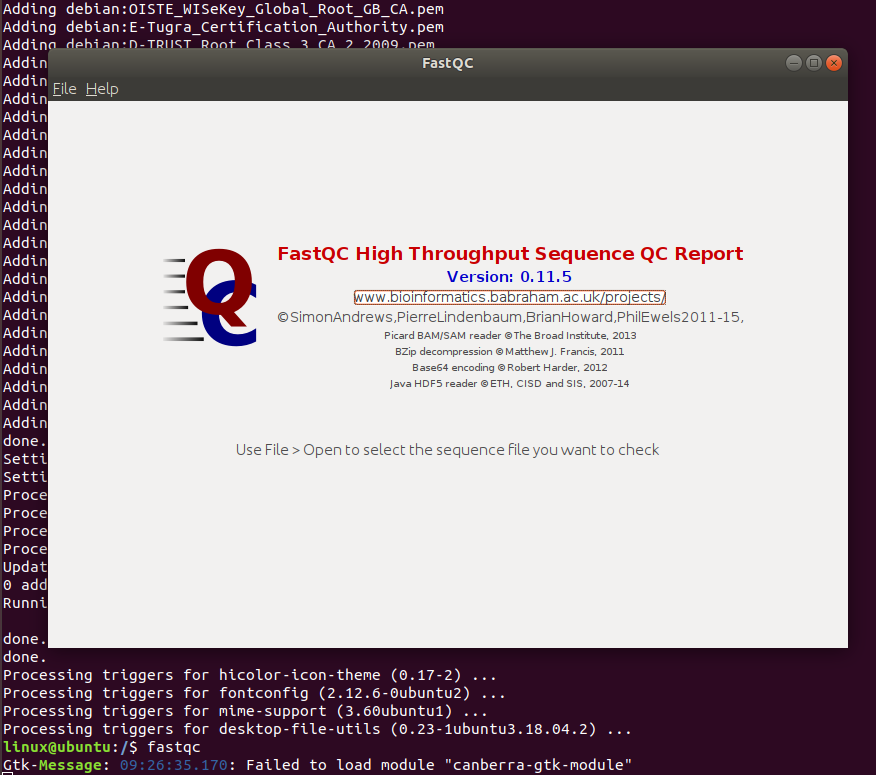
다시 깔기로 했다…….



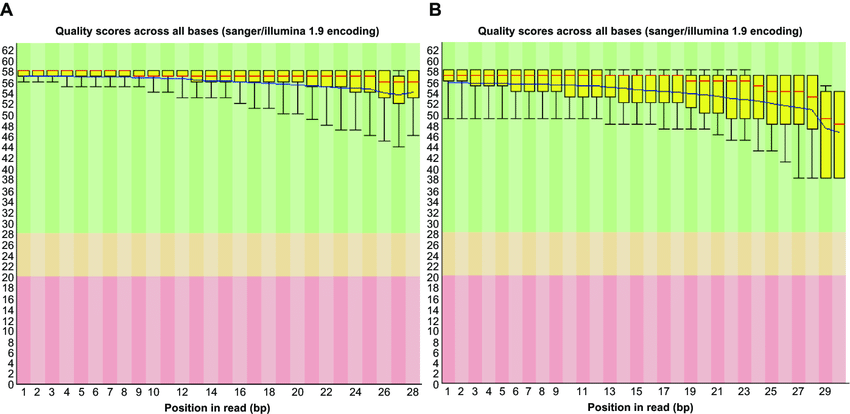
다시 권한을 부여하고 update하고 fastqc를 깔았다

이때 lock파일 때문에 오류가 났지만 rm으로 지웠더니 오류가 해결됐다. 이 lock파일은 업데이트를 막는 것이라고 한다. 그래서 오류가 났던 것으로 보통 세팅할 때 같이 설정되는 경우가 많아서 발생한 것으로 보인다.

아래는 실행시킨 화면이다.



데이터가 있다면 아래와 같은 화면을 확인할 수 있다.



출처 : <https://www.researchgate.net/figure/FastQC-analysis-of-ChIP-Seq-data-FASTQ-format-files-are-derived-from-short-read-NGS-data_fig1_236639867>

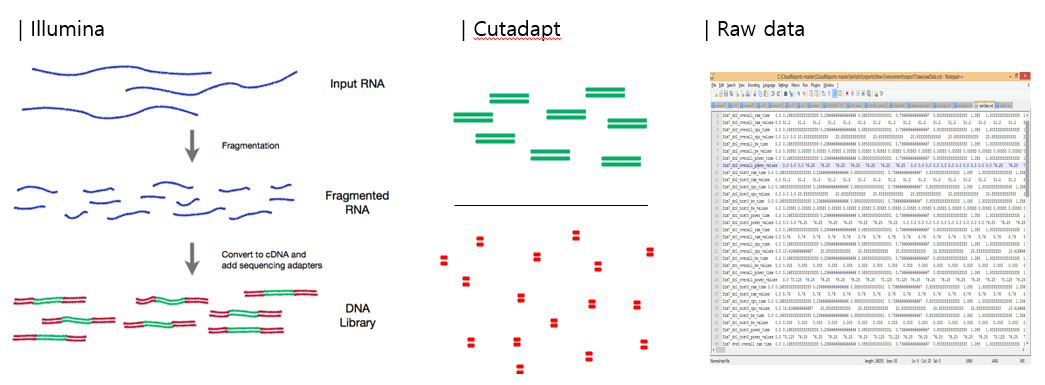
y축을 정확도라고 생각하면 된다.

Week3 – Cutadapt

이번 주에 드디어 데이터를 받았다. 이 데이터는 우리 실험실에 있는 van이라는 분의 것이다. 그 분은 ABC transport 중 어떤 것이 뿌리의 발달과 관계되는지 알아보기 위한 프로젝트를 진행하고 있다. 그 과정에서 이를 위한 후보군을 추리기 위해 RNA-seq을 부탁하셨다.

그래서 이를 위해 작업 공간을 VMware에서 xshell로 옮겼다. 왜냐하면 데이터의 크기가 80Mb라 너무 크기 때문이다. 또한 서버는 실험실에 있는 서버를 사용하고 있고 이때 코어의 개수는 총 8개다.

오늘은 이 다음 과정을 위해 Cutadpt를 깔아볼 것이다. Cutadpat에 대해 간략히 설명하자면 말 그대로 미처 잘리지 못한 adpat서열이 raw data 정보에 들어가 있을 때 이를 제거하는 과정이다. 이러한 과정이 필요한 이유는 fastqc에서 낮게 나온 정확도를 올릴 수 있는 하나의 방법이기 때문이다.



그렇다면Cutadapt가 무엇을 하는 지 알아봤으니 이번에는 어떤 명령인지에 대해 좀 더 자세히 살펴보자. cutadpat 명령은 pair end냐 single end냐에 따라 명령이 달라진다. 쉽게 말해 염색체를 쌍으로 sequencing 했다면 후자 일 것이고 single로 sequencing 했다면 전자 일 것이다. 따라서 명령 자체는 크게 다르지 않다. 내가 받은 파일은 이름만 보아도 \_1/\_2 로 pairend인 것을 짐작할 수 있다. 따라서 pair end의 명령만 간략히 보도록 하자.

cutadapt –a AGATCGGAAGAGC –g AGATCGGAAGAGC –q 숫자 –m 숫자 –o /home/경로1/파일이름.trimmed.확장자 –p /home/경로2/파일이름.trimmed.확장자 /home/경로3/파일이름 /home/경로4/파일이름

| 옵션

-a : 5’쪽 primer

-g : 3’쪽 primer

-q : quality (fastqc에서 확인했던 값/ 보통 30 이상으로 한다고 한다)

-m : 얼마 단위를 자를 것인지 (보통 20mer 정도로 맞춘다고 한다)

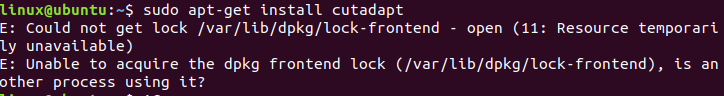
-o : output 파일의 경로를 지정하는 것이다. (경로1, 경로2 해당)

-p : input 파일의 경로를 지정하는 것이다. (경로3, 경로4 해당)

\*\*이때 실행하는 곳의 위치와 파일(input/output)위치가 같다면 생략하고 그 아래부터 적을 수 있다.

\*\*추가적으로 core를 사용한다면 -j 코어개수로 옵션을 줄 수 있다.

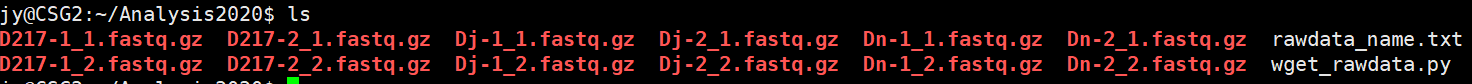
이제 cutadapt를 깔아보자. 그런데cutadapt를 깔려고 하자 저번주와 똑같은 오류가 또 발생했다.



저번과 같이 해결을 하려고 봤는데 update를 위해서는 다시 시작해야 한다는 창이 떠서 그렇게 했더니 해결은 됐다. 아마도 설치하기 전 sudo-apt get update를 안 해서 오류가 났었던 것 같은데 분명 저 오류는 설치를 할 때 setting 과정에서 발생한다고 했는데 정확한 이유를 모르겠다. 만일 다음 주에 sudo-apt get update 명령을 제대로 하고도 또 발생한다면 좀 더 자세히 알아봐야겠다.

(Xshell에서는 이미 cutadapt가 깔려 있기 때문에 까는 건 vmware에서 실제 작업은 Xshell에서 진행했다.)

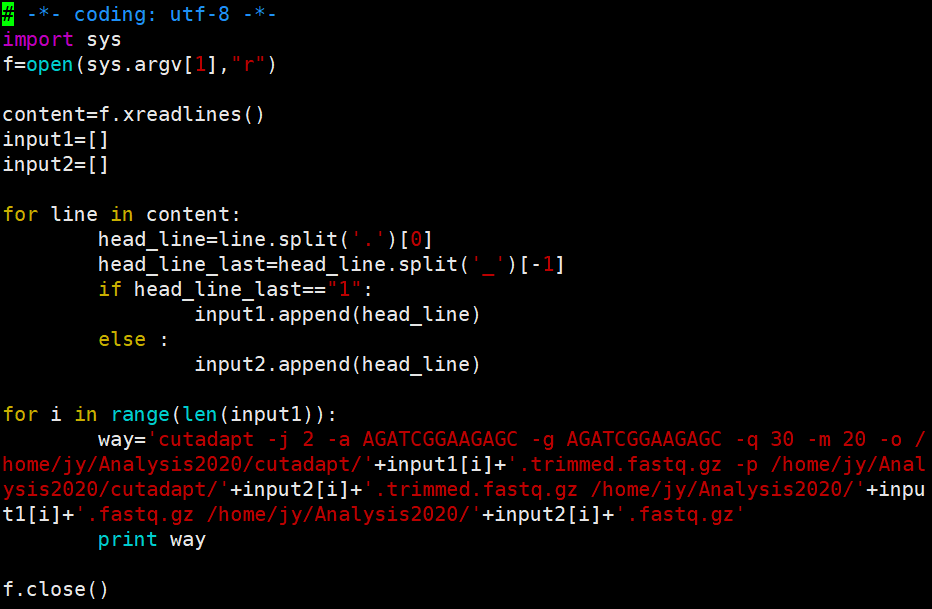
일단 앞 전의 오류는 update를 통해 해결한 것으로 하고 이 다음 과정을 진행해보기로 하였다. 먼저 내가 받은 파일은 총12개로 개개로 cutadpt 명령을 실행시키기에는 너무 오래 걸린다는 생각이 들었다. 이에 파이썬을 통해 자동으로 돌아갈 수 있도록 구현하였다.



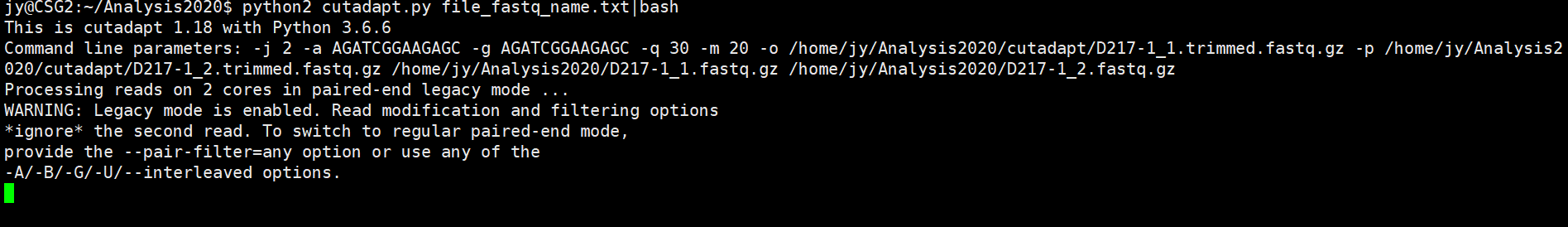
따라서 가장 먼저 grep으로 fastq가 들어있는 것들을 추려 하나의 파일로 만들었다.



다음으로 이렇게 만든 파일을 받아 cutadapt명령에 맞게 파싱한 후 print를 반복적으로 돌리는 python 파일을 만들었다.



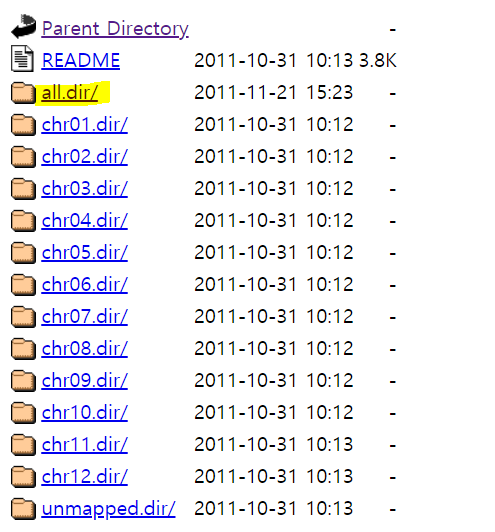
이를 단순히 실행하면 cutadpat 명령이 12개가 단순히 보여지고 전혀 실행되지 않는다. 이를 실행시키기 위해서는 이 것을 실행했던 결과 값을 bash로 넘겨줘야 했다. 이 명령은 어렵지 않았다.



이렇게 몇시간을 기다리면 cutadapt 가 정상적으로 작동하게 된다.

Week4 – gffread

rice genome에 대한 전체 서열 파일(.gff3)을 아래 코드를 통해 다운 받은 후 이름을 all.gff3라 명명하였다.

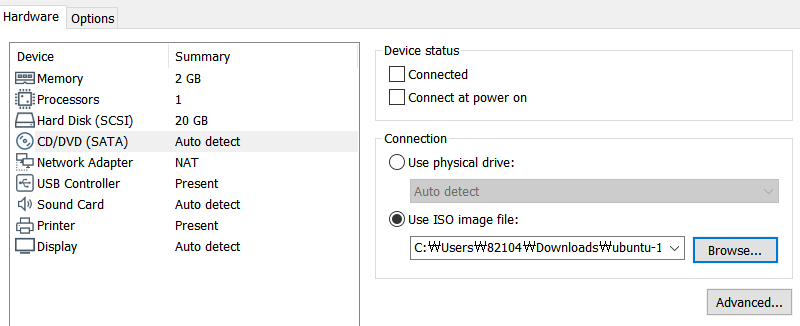


이후 이 파일을 exon 정보와 splice site 정보를 포함한 파일의 형태인 gtf 형식으로 변환시키기 위해 gffread를 다운 받아야 했다.

먼저 이를 다운받기 위해 sudo apt-get install gffread를 실행시켰지만 저번주와 같은 오류가 발생하였다. 문득 매번 같은 오류가 발생하는 데에는 어떤 다른 이유가 존재할 것이라 생각했다. 그렇게 다시 생각해보니 실행할 때마다 항상 아래와 같은 경고 창이 떴다는 것이 생각났고 이를 해결할 방법을 검색해보았다.

Cannot connect the virtual device sata0:1 because no corresponding device is available on the host

찾아보니 이는 virtual box가 연결돼 있는 곳이 cd/dvd로 설정 돼 있었기 때문에 발생한 문제로 보였다. 그래서 cd/dvd가 없는데 이를 여기에 연결하려고 하니 항상 오류가 생겼던 것 같다.

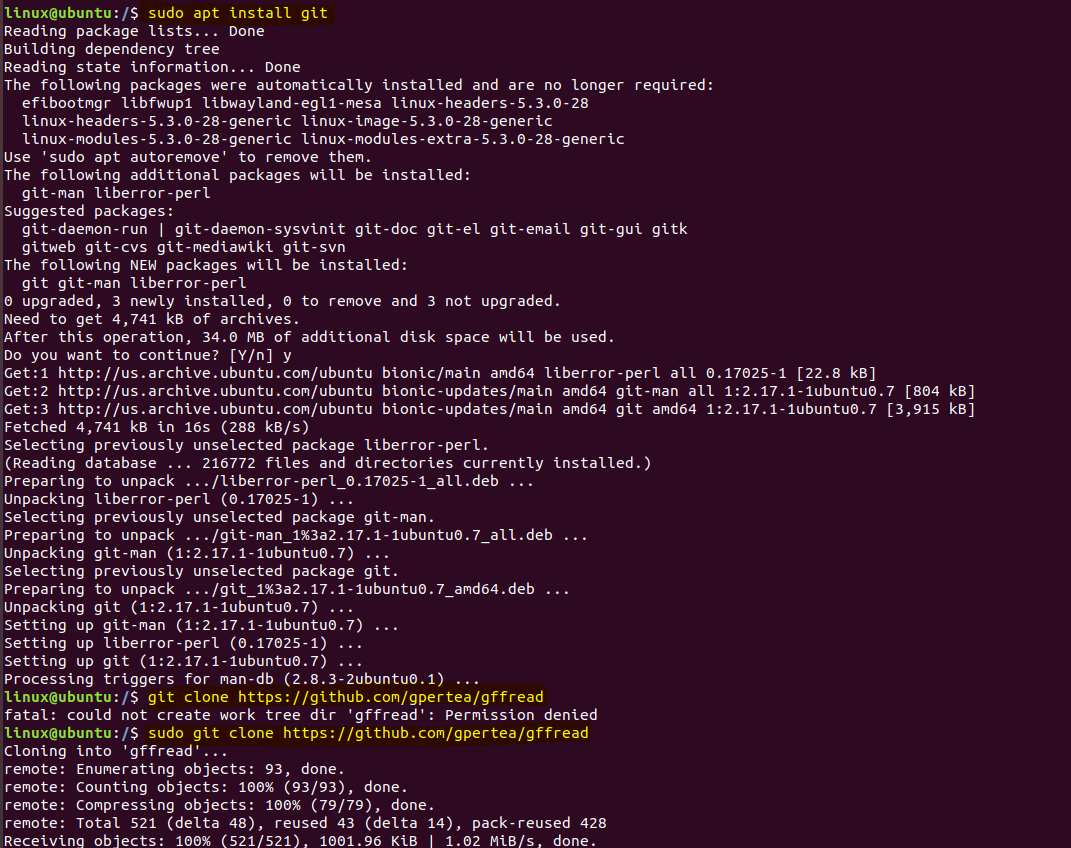


이 오류는 device status 부분을 선택하지 않고 use iso image file로 바꿈으로 써 생각보다 쉽게 해결할 수 있었다. 이렇게 하니 실행시에 더 이상 오류가 생기지 않았고 install을 할 때에도 저번주와 같은 오류가 발생하지 않았다.

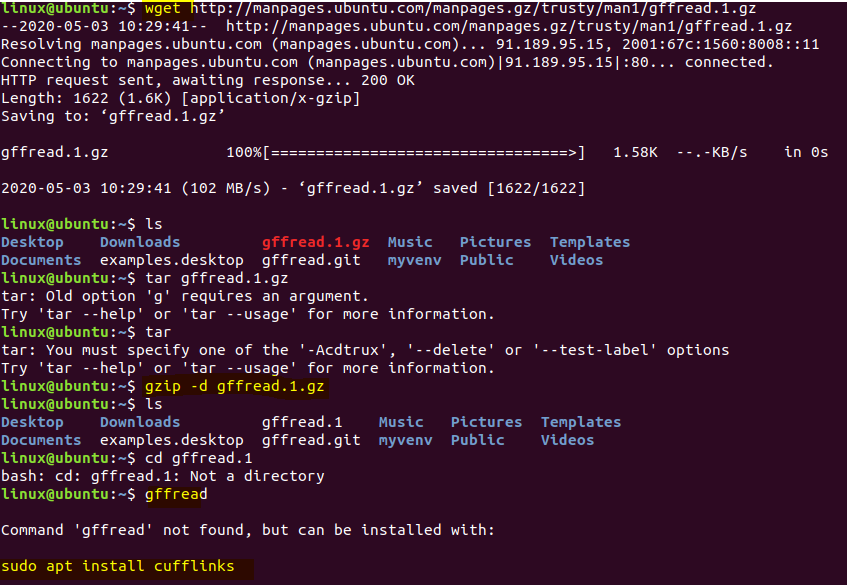
그런데 이번에는 다음과 같은 오류가 발생했다.

Unable to locate package gffread

이 오류를 해결하기 위해 또 한번 source.list 파일을 수정해 보기로 했다. 그런데 파일을 열어봤더니 더 바꿀 내용이 없었다. 이에 처음에는 update, upgrade의 문제라 생각하여 실행 후 해보았지만 여전히 같은 오류가 발생했고 git hub에 들어가서 wget을 통해 다운 받아 실행 시켜 보았지만 여전히 쓸 수 없는 형태였다. 그러다 [https://packages.ubuntu.com/eoan/gffread 이 곳에서 tar.gz](https://packages.ubuntu.com/eoan/gffread%20이%20곳에서%20tar.gz) 형태로 다운을 받게 되었는데 드디어 오류를 해결할 수 있었다. 그러나 명확한 원인은 파악하지 못하였다.



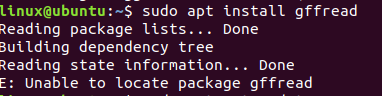
Git hub에 올라 와있는 gffread installation 과정을 따라한 것이다. 이를 위해 git을 깔고 git clone을 통해 다운 받았으나 여전히 오류가 발생하였다.



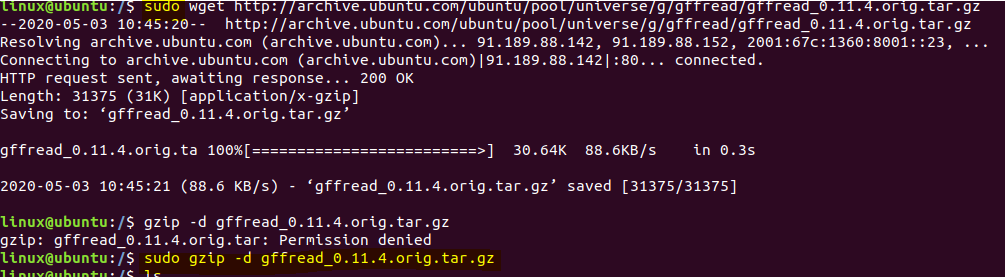
다른 사이트에 올라와있는 파일이였으나 역시 같은 오류가 발생하였다.

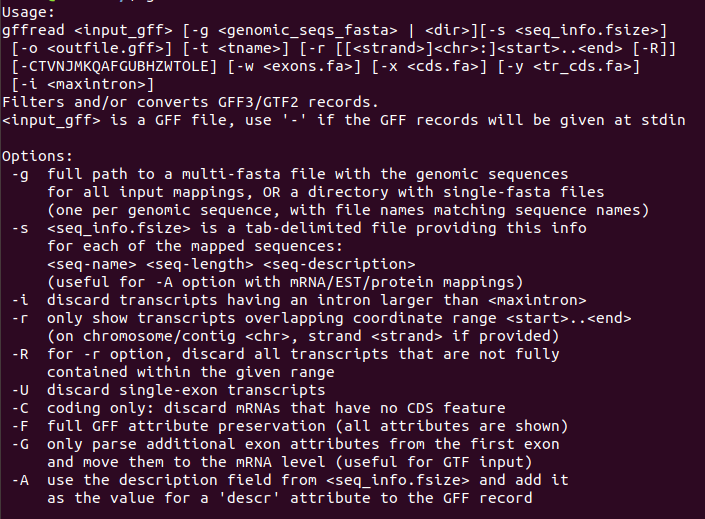


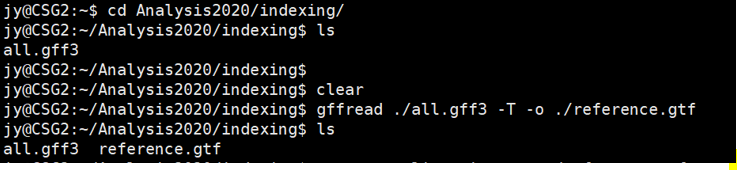
Gffread를 깔려고 하면 cufflinks를 깔라고 하는 말이 있어 깔아보았으나 아래와 같은 오류가 여전히 발생하였다.



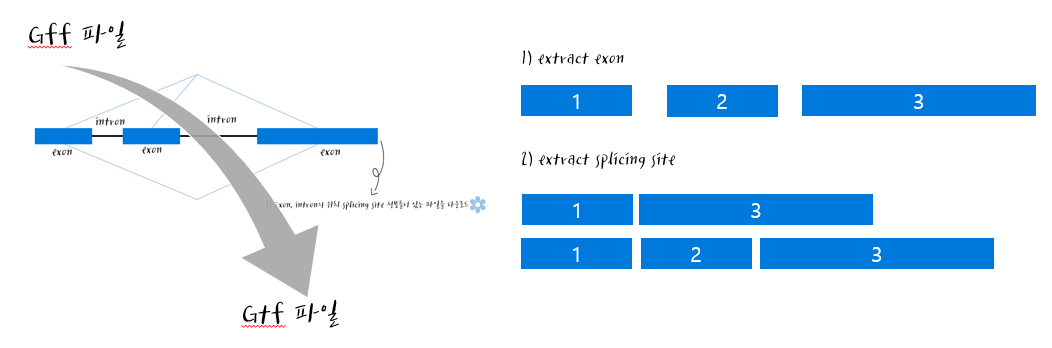
이에 위에 설명한 바와 같이 다운 하였더니 사용할 수 있게 되었다.



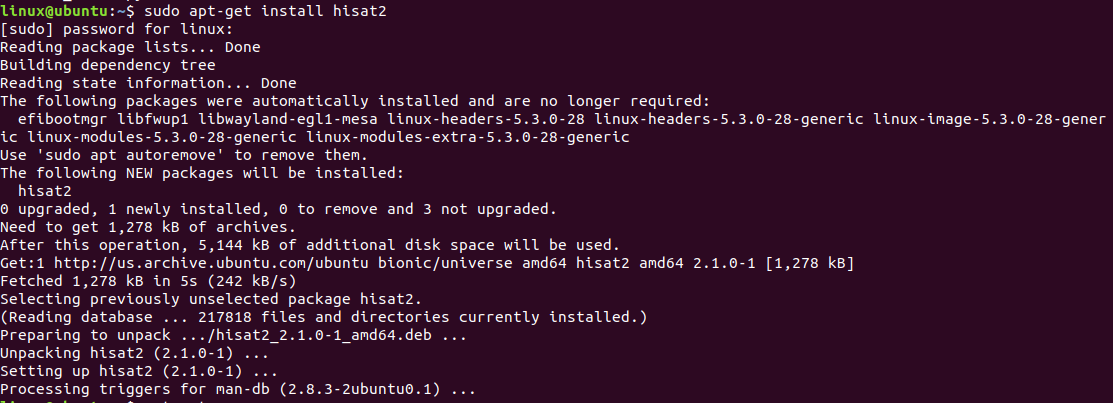


 xshell에서 실행시킨 화면이다

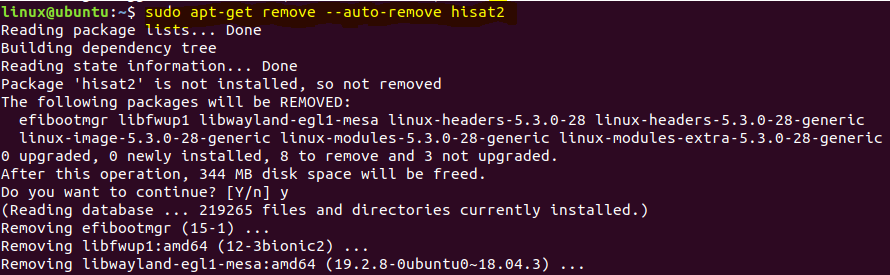
Week5 – indexing(extract splice site, extract exon, hisat2)



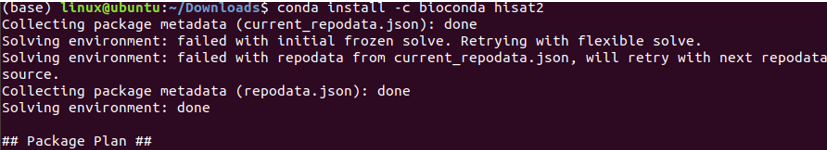
그 다음으로 진행해야 할 과정은 indexing이다. 이를 위해 hisat2를 다운 받아야한다. 다음을 통해 성공적으로 다운 받을 수 있었다

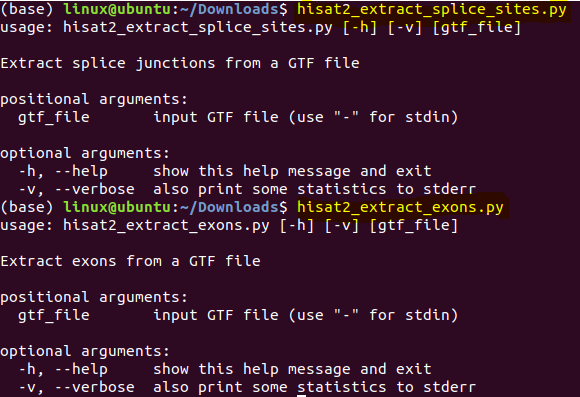


Hisat2 안에는 hisat2-build 명령이 들어가 있다. 그러나 extract를 하는 과정은 포함 돼 있지 않았다. 예전 버전에는 포함이 돼 있었던 것 같은데 2.1.0 버전에는 이 것이 포함되지 않은 것 같았다.



찾아보던 와중 conda 명령어로 설치를 하면 문제없이 실행된다고 해서 anaconda를 다운 받았다.



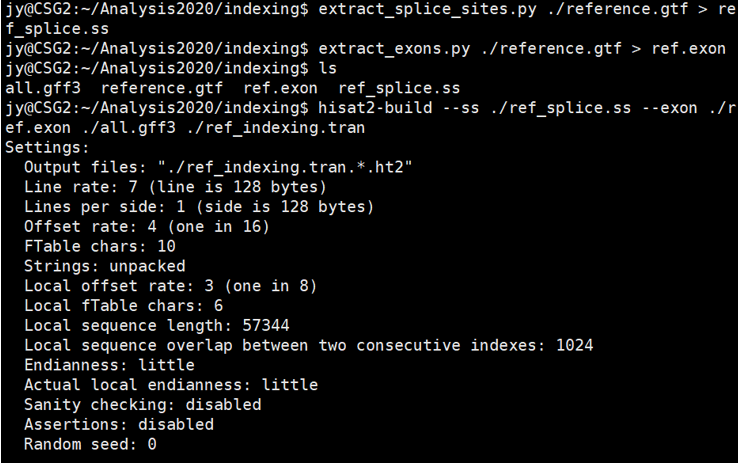


제대로 동작할 준비가 된 것을 확인할 수 있었다. 아래는 xshell을 통해 실행시킨 화면이다. 처음에 gff3파일을 써야하는데 gtf파일을 써서 오류가 발생했다.

아래는 오류가 발생했을 때의 화면이다.



제대로 완성시킨 모습은 아래와 같다.



그런데 문득 VMware에서 터미널 프롬프트 왼쪽에 (base)라는 게 왜 생겨났는지 궁금했다. 찾아보니 이는 anaconda가 자동으로 활성화 돼 있기 때문이었다. 이를 해결하려면 아래와 같은 명령어를 입력하면된다.

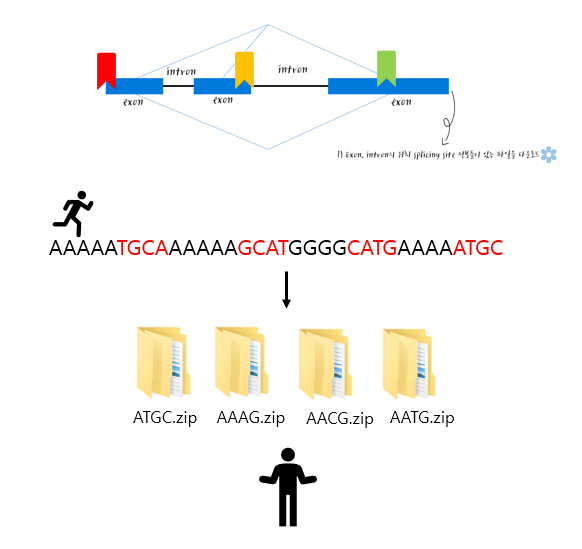
Conda config –set auto\_activate\_base False

Week6 – mapping/alignment(hisat2, samtools)

Indexing과 mapping은 사전에서 원하는 단어를 찾는 것과 그 과정이 매우 비슷하다. 예를 들어 coding이라는 단어를 찾는다고 가정해보자. 우리는 아래와 같은 순서에 따라 단어를 찾아낼 것이다.

1. c index를 찾는다.
2. 알파벳 순에 맞게 o,d,I,n,g를 찾는다.
3. 뜻을 알아낸다.

이때 1번 과정이 지난 주에 한 작업이고 이번주에 할 작업이 2번 과정이다. 이때 2번 과정은 굉장히 중요한데 그 이유는 다음과 같다.



먼저 이해를 돕기 위해 똑같이 coding이라는 단어를 찾는다고 생각해보자. 여기에 누군가가 첫글자가 똑 같은 거 끼리 모아놓은 박스에서 c 박스를 건네주며 coding이라는 단어를 찾으라고 요구한다면 어떨까? 이는 매우 귀찮고 까다로울 것이다. 더불어 이는 컴퓨터 입장에서도 마찬가지다. 매번 내가 찾고자 하는 게 어디 있는지 알아내기 위해 몇 만 개의 글자를 일일이 비교한다는 것은 매우 소모적이다. 이에 우리는 이것들을 일정한 규칙에 따라 압축해 놓았고 그것과 관련된 파일만 압축을 품으로써 컴퓨터의 성능을 올릴 수 있었다. 이번주에는 이 과정을 진행할 것이고 우리는 이를 위해 hisat2와 samtools라는 도구가 필요하다.

운이 좋게도 우리는 저번 주 indexing 과정을 위해 hisat2 package를 다운 받았다. 그렇기 때문에 이번주에는 samtools만 추가적으로 다운 받으면 된다. 코드는 아래와 같다.

Sudo apt-get install samtools

정상적으로 다운 받은 후 hisat2를 통해 먼저 sam 파일을 만들어주었다. sam파일을 만드는 코드는 아래와 같다.

hisat2 –p 8 --dta –x tran파일저장경로/파일이름\_tran -1 1번파일경로/파일이름 -2 2번파일경로/파일이름 –S 파일저장경로/파일이름.sam

| 옵션

-p : thread 개수

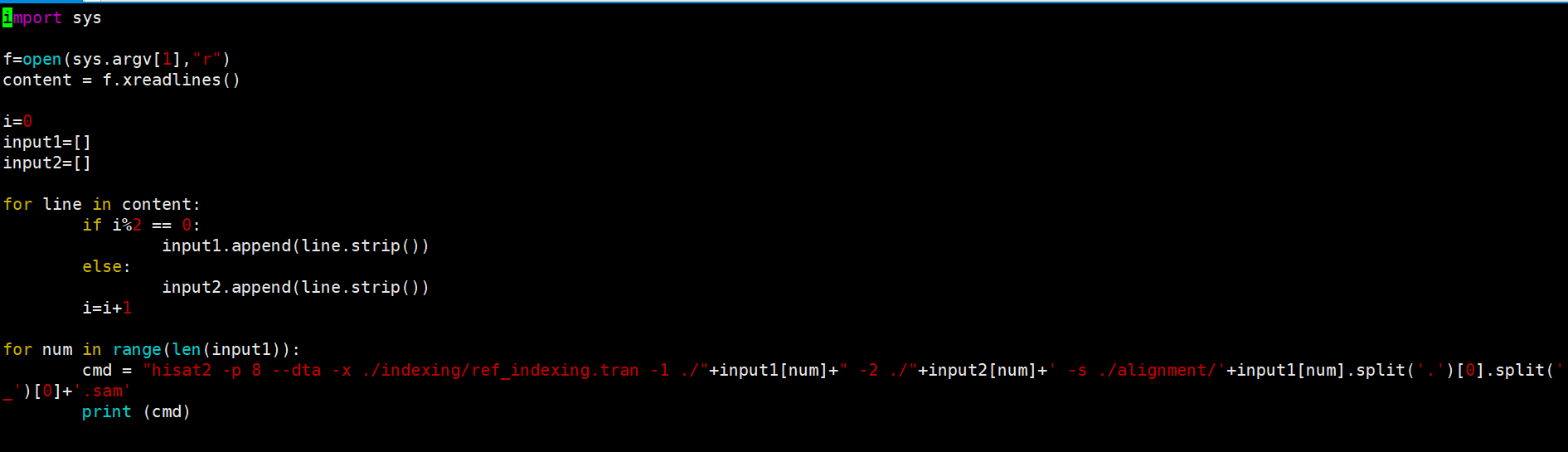
--dta : transcript assemblers에 맞게 조정된 alignment들을 보고함

-x : indexing을 통해만든 tran 파일

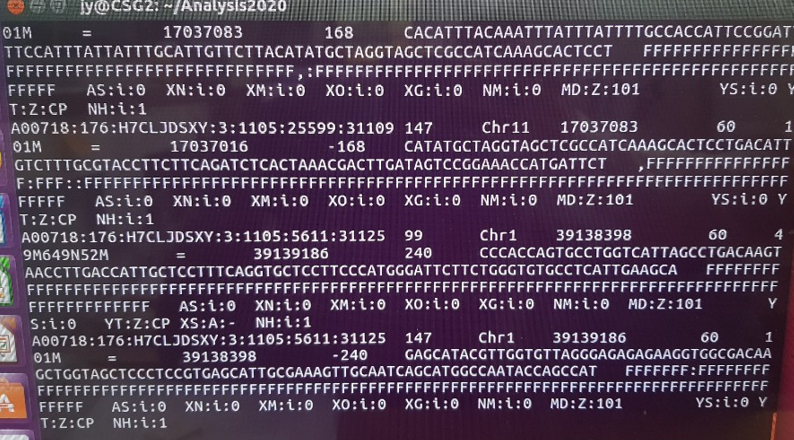
-S: 결과 파일 저장

이 일 또한 12개의 파일 하나씩 하나씩 바꿔주는 것은 매우 귀찮기 때문에 hisat2.py라는 파일을 만들어 한꺼번에 진행하였다.

| hisat2.py



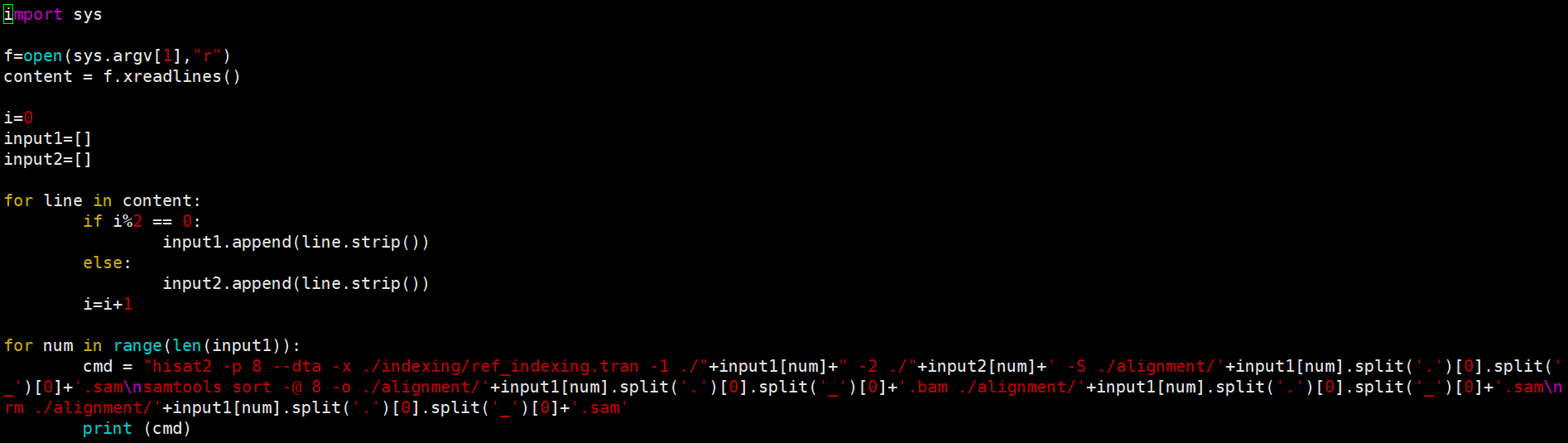
서버에서 돌린 모습은 아래와 같다.

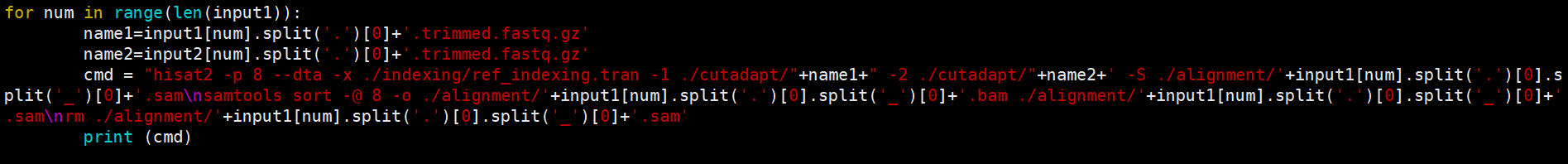
  
이때 까지만 해도 이 모습이 굉장히 빠르게 잘 돌아가는 모습인 줄 알았다. 그러나 이는 옵션을 잘못 준 것이었다. -S option을 소문자 s로 표기하였다. 소문자 s는 input으로 들어온 파일의 첫번째 내용을 skip하라는 것이었다. 그래서 화면에 보이는 모습은 skip한 내용을 나타내고 있는 듯 하다. 그래서 뒤늦게 이를 발견하고 option을 변경하였으나 이번에는 공간이 부족해서 에러가 났다.

그래서 다음과정인 samtools가 결국 bam파일을 압축하여 sam파일을 만드는 것이기 때문에 이 과정까지 한꺼번에 진행하기로 하였다. Samtools를 사용하는 방법은 다음과 같다.

samtools sort -@ 8 –o 결과파일경로/파일이름.bam sam파일저장경로/파일이름.sam

| 수정한 hisat2.py

순간적으로 용량을 높이기 위해 fastq.gz 파일을 지우기로 하였다. 그런데 이걸 지울 이유가 없었고 그래서 위 파일이 돌아가지 않았다. 그런데 다시 생각해보니 fastq.gz 파일이 아닌 trimmed 한 파일을 써야 한다는 것을 잊고 있었다. 그래서 ‘cmd’부분만 아래와 같이 수정하였다.

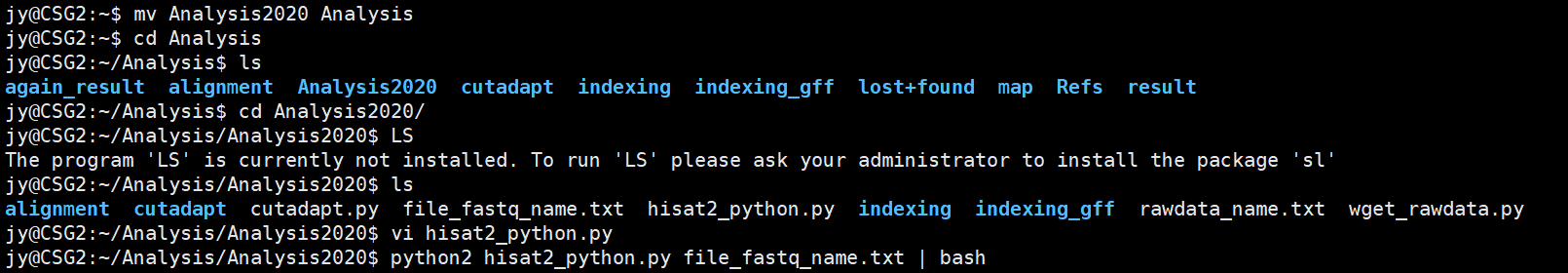


그러나 또다시 용량 부족의 늪에 빠지게 되었다.

텍스트이(가) 표시된 사진

자동 생성된 설명

그래서 Analysis라는 directory를 만들고 여기에 할당된 메모리를 높였고 기존의 모든 내용을 이 directory로 옮겼다.



그랬더니 성공적으로 파일이 만들어지기 시작했다. 다음부터는 명령어를 콘솔에서 실행하기 전에 명령어 자체에 대해 더 공부한 후 돌려야겠다고 다짐했다. 이 일로 며칠을 날렸는지 모르겠다.

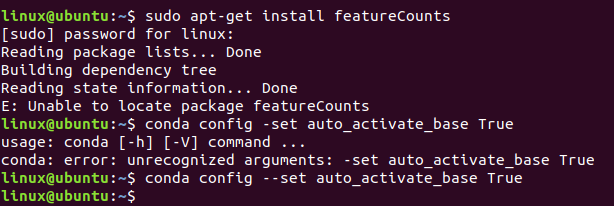
텍스트, 전화, 휴대폰, 앉아있는이(가) 표시된 사진

자동 생성된 설명

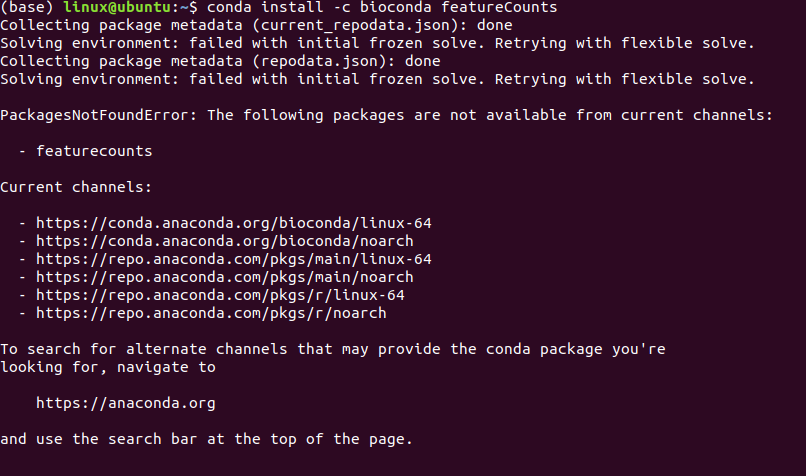
서버에서 정상적으로 돌아가고 있는 모습이다.

Week7 – counting(featureCounts)

이번주에는 RNA-seq의 마지막 단계인 counting을 진행할 것이다. 저번주까지는 내가 가진 dna 조각들이 reference에 어느 위치에 일치하는지를 찾아냈기 때문에 이번주에는 단순히 그 개수를 세는 과정을 진행할 것이다. 이때 필요한 tool은 featureCounts 하나이다.



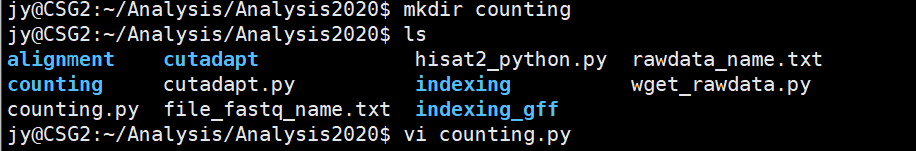
Sudo apt-get install featureCounts 로 다운 받고 싶었으나 실패하였다. 이에 다시 anaconda를 활성화 시켜 이곳에서 다운 받아 보기로 하였다.

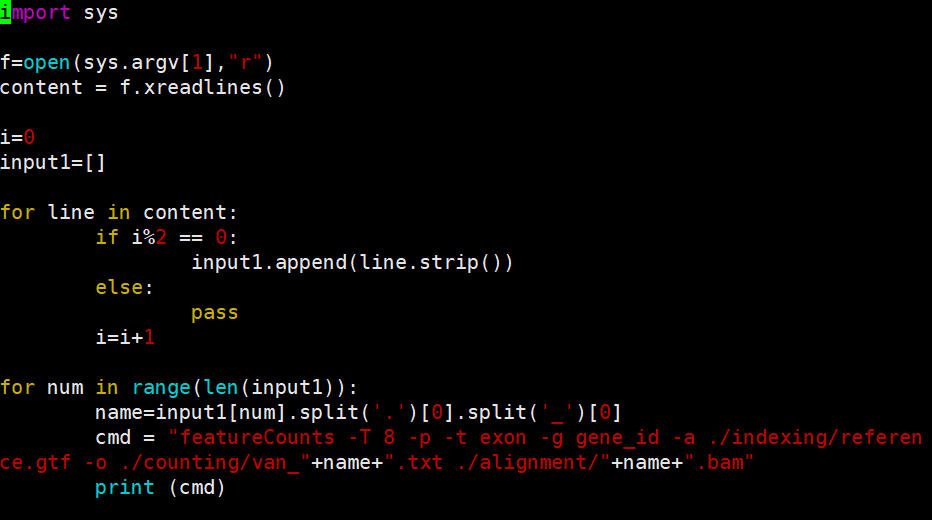


이렇게 해서 성공적으로 설치를 마쳤다.

| code

featureCounts –T 2(서버숫자) –p(paired end라서) –t(exon을 default로 줌 gtf 타입결정) exon –ㄴg gene\_id –a gtf파일경로/파일이름.gtf –o(output) 결과파일저장경로/파일이름.txt bam파일경로/파일이름.bam





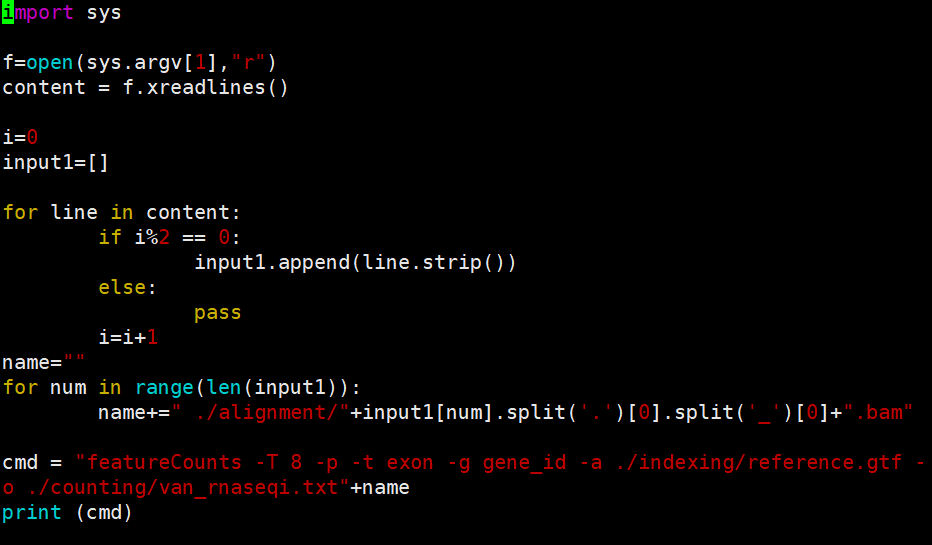
이렇게해도 문제는 없으나 이렇게 할 경우 하나의 파일로 요약되는 것이 아니라 각각의 파일로 쪼개진다. 그런데 이후에 R을 통해 deg를 뽑는 작업을 할 때 하나의 파일로 구성된 것이 작업 하기 편하다. 그래서 하나의 파일로 작성하기위해 다시 내용을 수정하였다.

아래 화면은 하나의 파일로 돌아가는 모습이다.(서버화면)

텍스트, 전화이(가) 표시된 사진

자동 생성된 설명

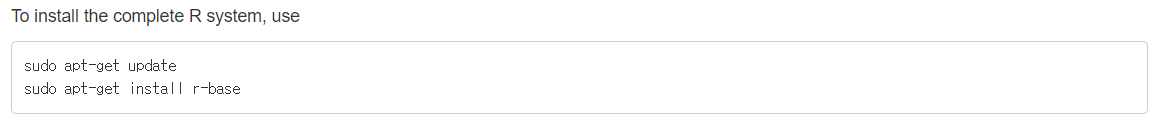
아래와 같이 수정하였고 하나의 rnaseq파일로 잘 만들어졌다.

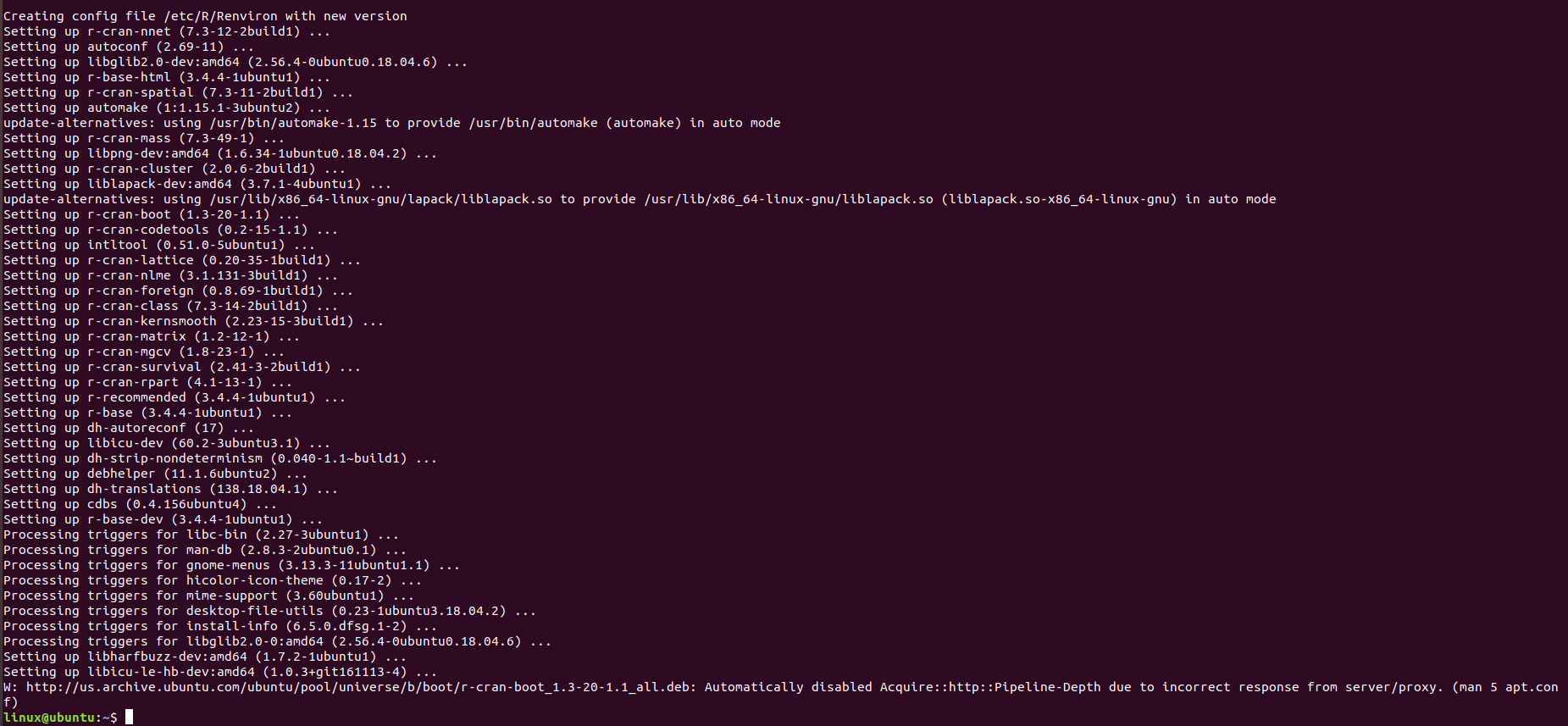


Week8 – visualization(R studio)

리눅스에 R과R-studio를 설치하고 이를 이용해 저번 주에 만든 counting 파일을 시각적으로 표시해 보자.

먼저 <https://cran.r-project.org/> 로 들어가 linux – ubuntu 버전에서 설치하는 법을 확인해 보니 다음과 같았다.





꽤 오래 걸린 후 정상적으로 설치 되었다. 이제 R-studio를 깔아보자

찾아본 결과 R-studio를 설치할 때 에러가 안 나기 위해서는 jpeg runtime library와 GStreamer librariy를 설치해야 했다.

| code

 sudo apt install libjpeg62

wget --tries=3 --timeout=120 [http://ftp.ca.debian.org/debian/pool/main/g/ gstreamer0.10/ libgstreamer0.10-0\_0.10.36-1.5\_amd64.deb](http://ftp.ca.debian.org/debian/pool/main/g/%20gstreamer0.10/%20libgstreamer0.10-0_0.10.36-1.5_amd64.deb)

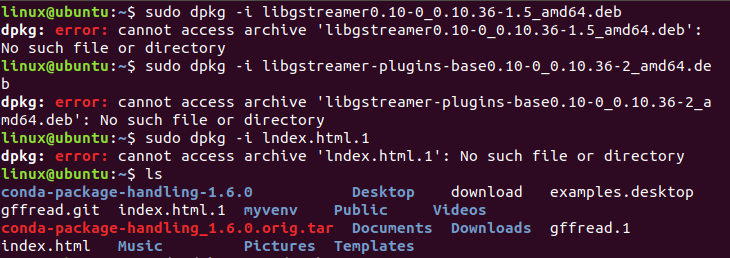
wget --tries=3 --timeout=120 [http://ftp.ca.debian.org/debian/pool/main/g/ gst-plugins-base0.10/libgstreamer-plugins-base0.10-0\_0.10.36-2\_amd64.deb](http://ftp.ca.debian.org/debian/pool/main/g/%20gst-plugins-base0.10/libgstreamer-plugins-base0.10-0_0.10.36-2_amd64.deb)

~~sudo dpkg -i libgstreamer0.10-0\_0.10.36-1.5\_amd64.deb~~

~~sudo dpkg -i libgstreamer-plugins-base0.10-0\_0.10.36-2\_amd64.deb~~

~~sudo apt-mark hold libgstreamer-plugins-base0.10-0~~

~~sudo apt-mark hold libgstreamer0.10~~



위에서 wget 까지는 정상적으로 작동했지만 아래부터는 되지 않았다.

그래서 GStreamer 1.0 을 개발환경으로 맞추려고 시도했다. (처음에는 저 버전이 사라진 줄 알았다)

| code

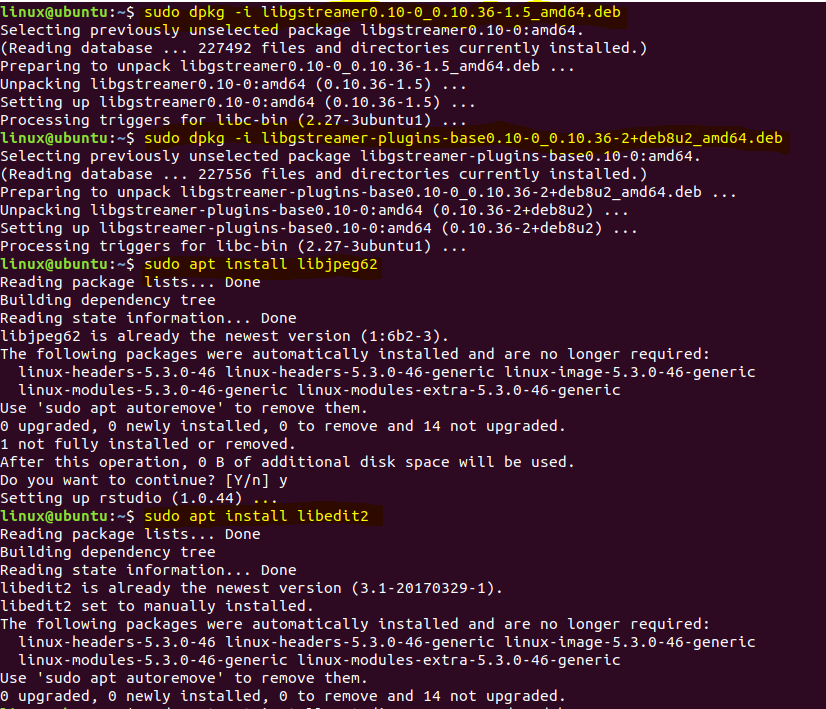
~~sudo apt install libgstreamer1.0-0~~

~~sudo apt install gstreamer1.0-libav gstreamer1.0-doc gstreamer1.0-tools~~

~~sudo apt install gstreamer1.0-plugins-base gstreamer1.0-plugins-good gstreamer1.0-plugins-bad gstreamer1.0-plugins-ugly~~

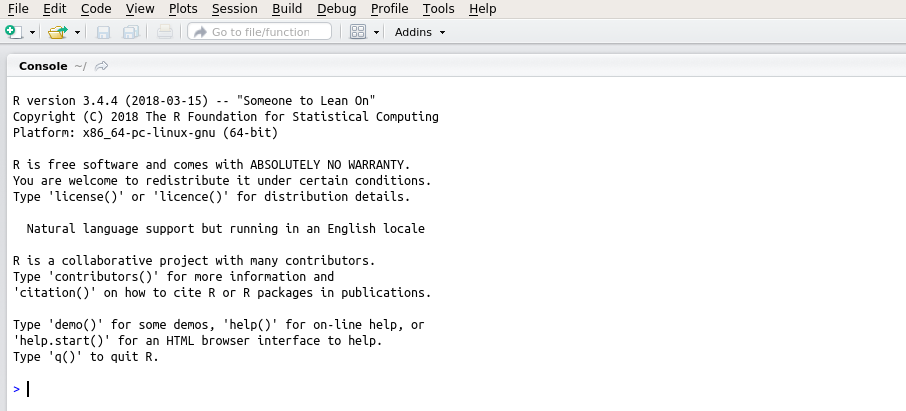
그런데 이 과정 이후에 r studio를 설치할 때 결국 필요한 건 gstreamer0.10 ~ 이라는 것을 알게 되었다. 그래서 이 오류를 해결하기위해 열심히 찾아 보았다.

정확하지는 않지만 ubuntu의 버전별로 설치방법이 조금씩 달랐고 이에 따른 오류라고 생각되었다.



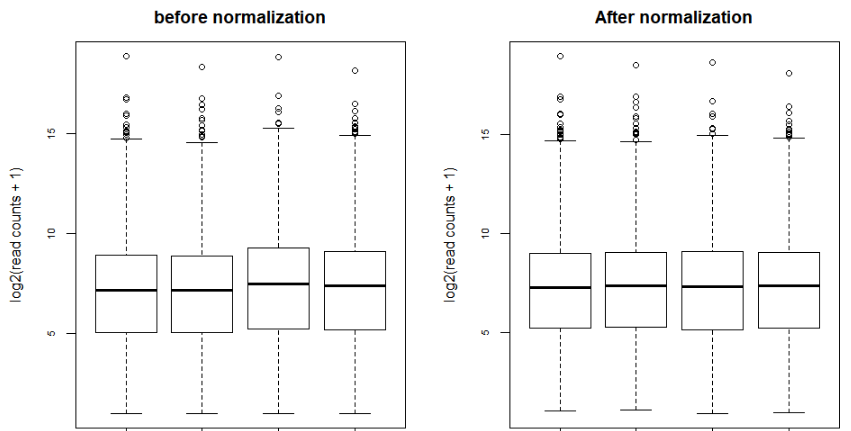
따라서 위와 같은 방식으로 설치하게 되었다.

이후 sudo dpkg -i rstudio-1.0.143-amd64.deb 로 rstudio를 설치해주면 성공!



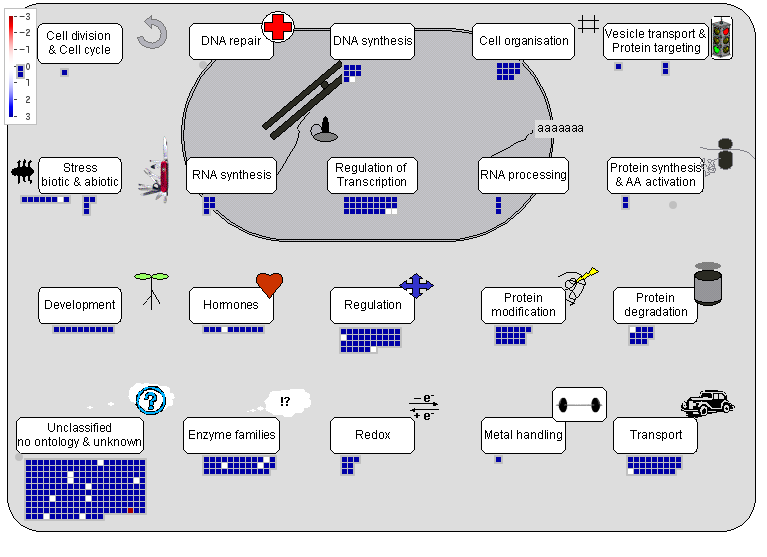
Week9 – visualiztion(mapman)

저번주는 R을 통해 아래와 같은 figure를 그릴 수 있었다.

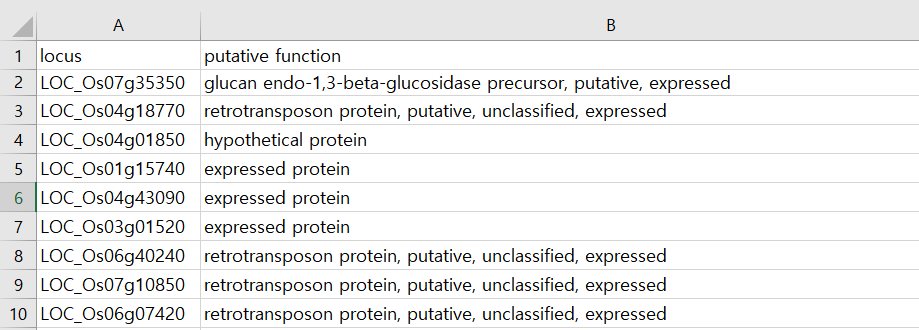


이번주에는 내가 가진 유전자들이 생체내의 어떤 system에 부합하는지 알아보자

예를 들어 내가 가진 유전자가 stress를 받을 때 활성화되는 유전자라면 그때 우리는 stress pathway위에 위치한 유전자를 mapman이란 도구를 통해 모식도로 확인할 수 있다.

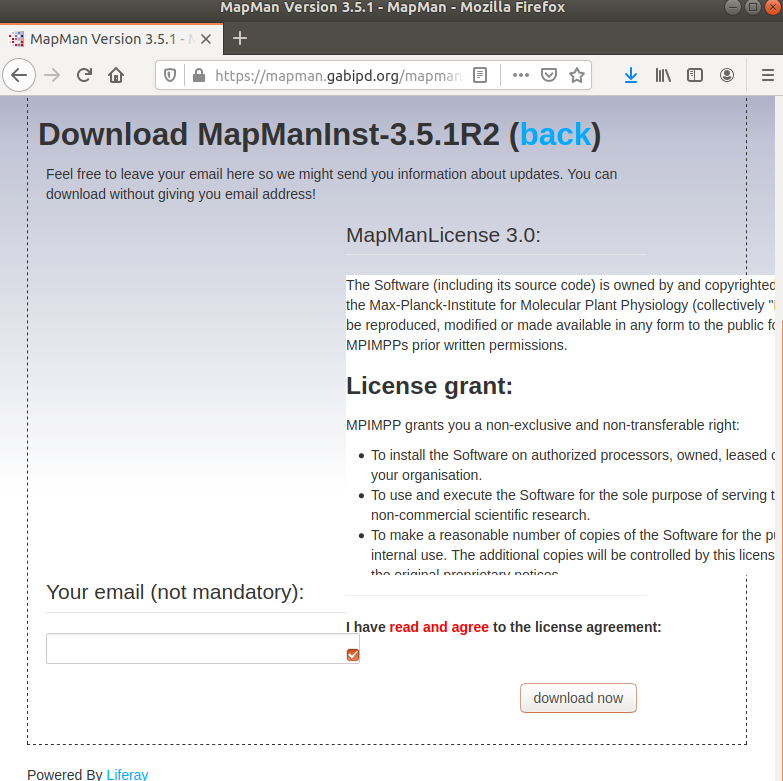
 

물론 굳이 이를 사용하지 않더라도 이 유전자가 어떤 역할을 하는지 나타내주는 사이트가 있지만 이를 통해 보는 것은 매우 비효율 적일 것이다.



왜냐하면 보통 하나의 분석을 하기위해서는 우리가 찾아봐야 할 유전자가 만개 이상이고 이를 하나하나 글로 읽는 다는 것은 결코 쉬운 일이 아니기 때문이다.

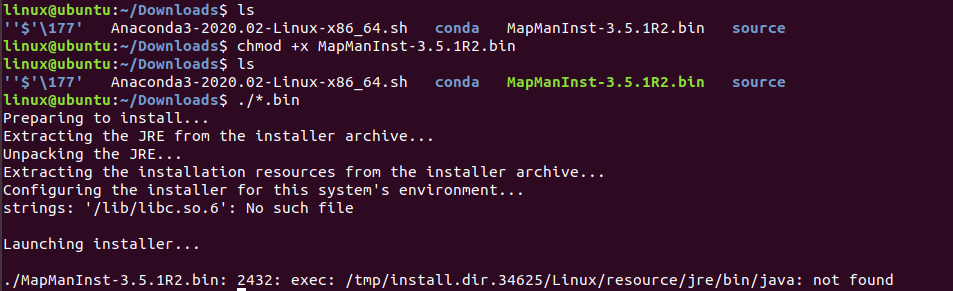
다운을 위해 mapman site로 들어가봤는데 링크 주소를 활용해서 다운 받는 방법이 보이지 않았다. 그래서 firefox를 이용해 bin파일을 다운 받았다.



리눅스에서 bin 파일에 실행가능 속성을 부여하고 설치하는 법은 다음과 같다.

| code

chmod +x 파일명.bin 한다음에 sudo (경로)파일명.bin



그런데 위와 같은 오류가 발생하였다.

찾아보니까 다른 사람도 mapman을 ubuntu에 설치하려고 했으나 이 단계에서 애를 먹고 있었다. 거기서 shell script를 사용해서 시도해보라고 했지만 여전히 오류가 해결되지 않았고 다른 사람도 마찬가지 였다. 그러나 그 분은 내가 깐 버전을 설치함으로써 더 이상 오류가 생기지 않고 바로 실행이 되었다고 한다.

예상되는 이유를 ‘looking back at it, I would guess it's a problem if you use the java virtual machine which comes with the installer or to use the java virtual machine which comes with the operating system. Probably the one which came with the installer did not resolve the needed system libraries on your linux installation.’ 이렇게 설명하셨는데 정확히 어떤 뜻인지는 와 닿지 않았다. Install 과정에서 자바가 문제가 되는 것 같아 이를 따로 깔아주고 실행시키면 되지 않을까란 생각이들어 자바를 깔아 보기로 했다.

그러나 자바를 성공적으로 깔았음에도 불구하고 오류는 해결되지 않았고 혹시나 하여 구버전을 깔아보았지만 설치되지않았다. 또한 버전별로 linux용이 없는 것도 많았기 때문에 mapman 자체가 linux에 적합하지 않은 프로그램일 것으로 예상되었다. 더불어 나와 정확히 같은 오류가 발생한 분은 저 글 하나 말고는 없는 것으로 보아 ubuntu위에 mapman을 까는 일은 별로 없는 것 같다. 아쉽게도 이번 작업은 window에서 진행하게 될 것 같다.

추가적으로 자바를 설치하는 명령은 다음과 같다.

sudo apt-get install openjdk-8-jdk

week10-data analysis tool(rapid miner)

저번주까지 해서 van의 프로젝트를 위한 figure는 모두 만들어서 van에게 넘겼고 성공적으로 후보군을 좁힐 수 있었다. 물론 지금까지 보여준 방법 외에도 다른 tool들을 이용해 heatmap을 그리거나 다른 figure들을 만들었지만 linux에서 돌아가지 않는 프로그램들이 많아서 window에서 작업을 하고 넘겼다. (혹은 단순히 r이나 엑셀을 사용하는 작업들이 많아서 생략하였다.)

그래서 이번주부터는 data analysis에 사용되는 다양한 tool들을 설치해보고 경험해 보려고 한다. 처음 접해볼 것은 rapid miner다.

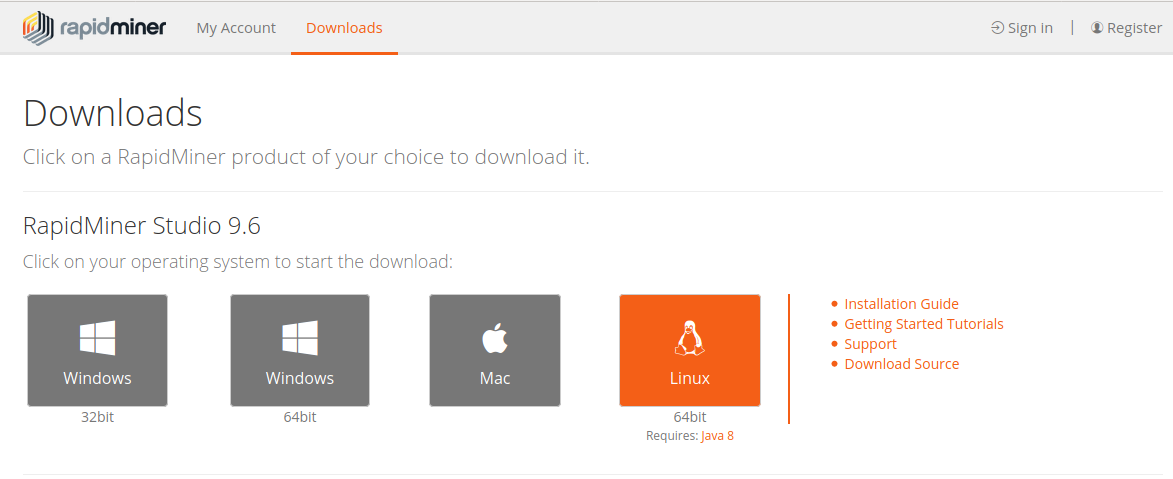
Rapidminer는 데이터 분석을 쉽게 해주는 툴로써 데이터전처리, 머신러닝, 데이터 모델링 등 데이터 분석에 필요한 모든 단계를 제공한다. 이러한 rapidminer에서 제공하는 tool은 총 5가지가 있는데 그 중에서 RapidMiner Studio가 가장 유명하다.

그렇다면 이러한 tool은 왜 필요할까? 예를 들어 우리회사에서 데이터 분석을 해야 한다고 생각해보자. 알다시피 데이터 분석은 일의 정확한 분담도 힘들고, 어떤 단계를 거쳐 분석이 행해졌는지 한 눈에 파악하기도 힘들다. 때문에 이러한 일들을 보기 쉽게 정리해 줄 도구가 필요하게 되었고 이를 가능하게 한 것이 바로 rapidminer이다.

뿐만 아니라 rapidminer는 local PC에 있는 파일부터 DB에 있는 파일까지 모든 파일을 자유롭게 import 할 수 있고 간단한 수치의 통계적인 정보나 이를 visualization하는데 특화 돼있다. 더불어 클러스터링, 회귀 분석, 시계열 분석, 딥러닝 등 다양한 supervised learning과 unsupervised learining 알고리즘을 제공한다는 장점을 가진다.

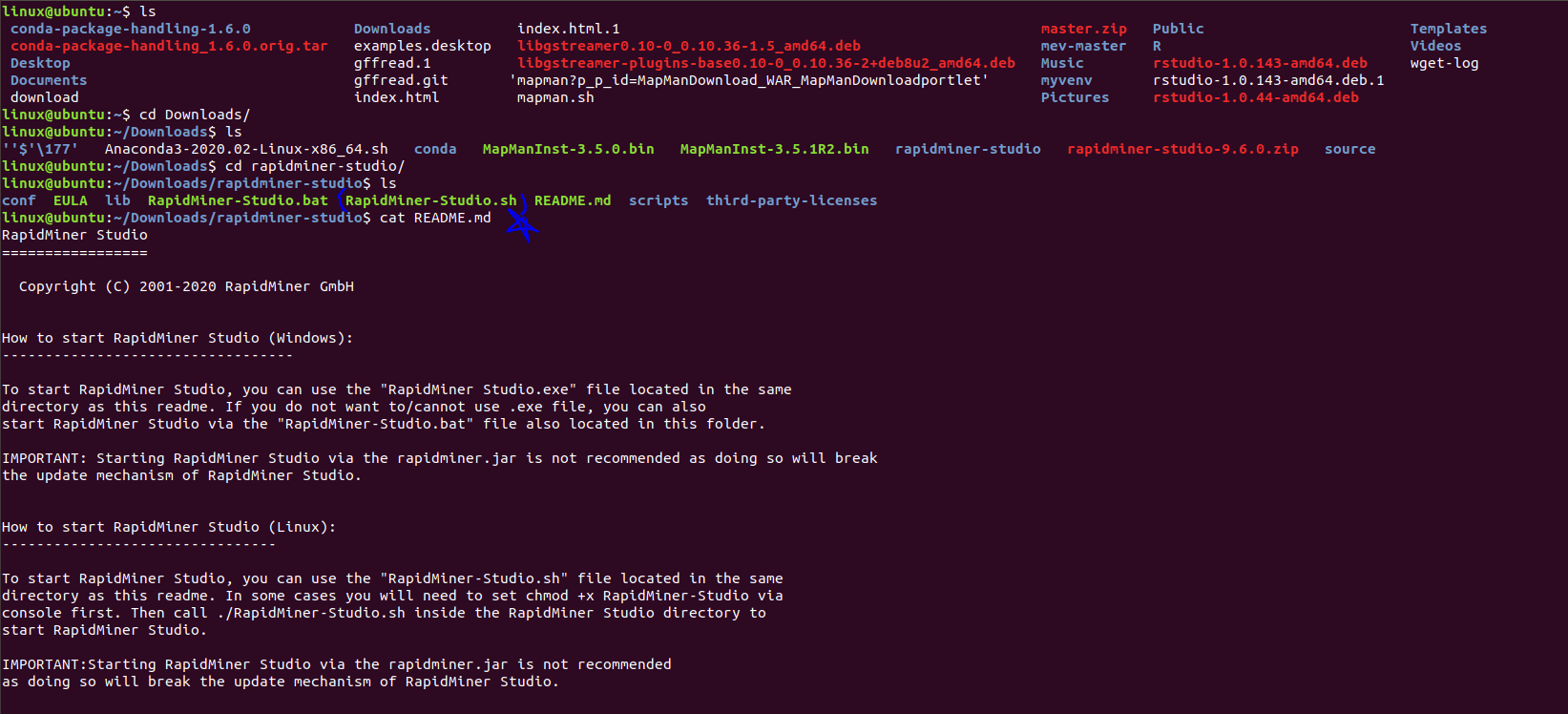
그렇다면 지금부터 ubuntu위에 rapidminer를 설치해보자

먼저 rapidminer의 공식 홈에 들어가 linux용을 다운받는다.

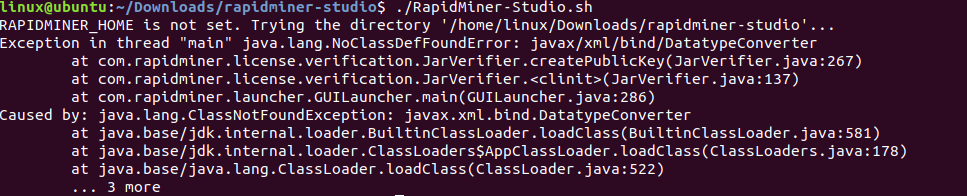


cd Downloads/rapidminer-studio

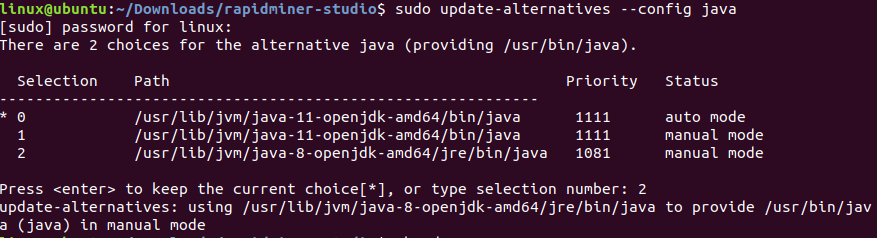
sh Rapidminer-Studio.sh 로 실행시킨다.



그런데 아래와 같이 제대로 실행이 되지 않았다.



이러한 오류가 뜬 이유는 이 프로그램을 깔 때 java8이 사용되기 때문이다. 돌이켜보니 공식 홈에 java8을 사용한다고 표기돼 있었는데 깜빡했다. 그래서 아래의 code를 이용해 사용되는 java mode를 바꿔 주었다.



이후 다시 sh 파일을 실행시켜 성공적으로 설치할 수 있었다.



Week11-data analysis tools(orange)



이번주에는 또다른 data analysis tool인 orange를 설치할 것이다.

Orange는 data mining에 특화된 도구로 아래와 같은 다양한 특징을 가지고 있다.

1. Interactive data visualization

통계 분포, box plot 및 산점도를 탐색하거나 의사 결정 트리, 계층 적 클러스터링, 히트 맵, MDS 및 선형 투영법이 가능하다.

1. Visual programming

GUI를 사용해 coding대신 데이터분석에 집중할 수 있도록 했으며 default를 통해 prototyping을 매우 빠르게 할 수 있다. 또한 캔버스 위에 위젯을 두고 서로를 연결함으로써 손쉽게 데이터를 다룰 수 있다.

1. Teachers and students love it

학교, 대학교 및 전 세계의 전문 교육 과정에서 사용되며 데이터 과학의 실습 교육 및 시각적 개념 삽화를 지원한다. 또한 가르치기 위해 특별히 설계된 위젯까지 있다.

1. Add-ons extend functionality

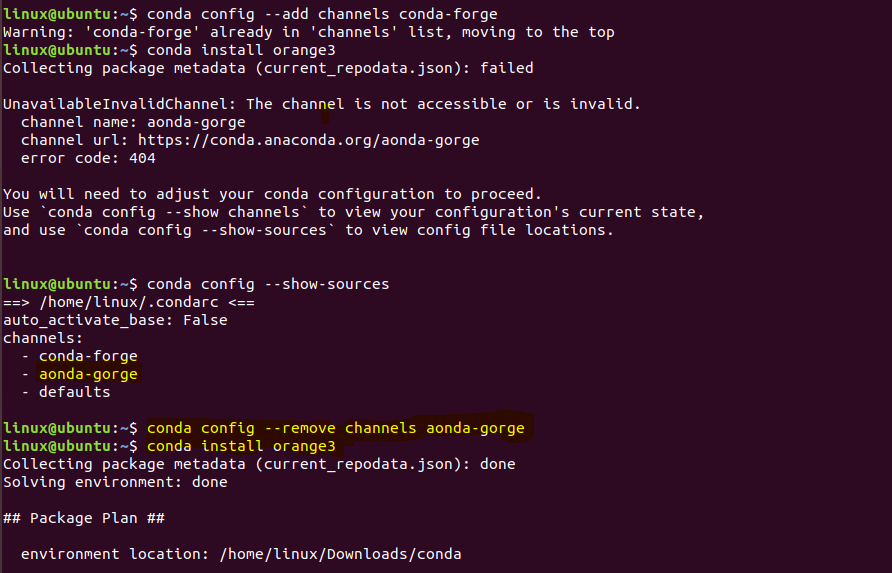
다양한 애드온을 사용하여 외부 데이터 소스에서 데이터를 채굴하고 자연어 처리 및 텍스트 마이닝, 네트워크 분석, 연관 규칙 마이닝이 가능하다. 또한 차등 발현으로 유전자의 순위를 정하고 농축 분석을 수행 할 수 있다.

그리고 당연히 나의 눈길을 끈 것은 bioinformaticians 들을 위한 add-on 기능이 있다는 것이었다. 그래서 오늘은 orange를 깔고 add-on 기능을 활용하여 scOrange를 추가로 설치하는 것까지 진행할 것이다.

\*\*scOrange란?

이것의 full name은 single-cell add-on for Ornage로 bioinformaticians들을 위한 tool이다. 이는 수천개의 유전자와 세포의 특징, pathway 등을 활용해 indivisual cell의 내부의 작용이 어떤지 예측해준다.

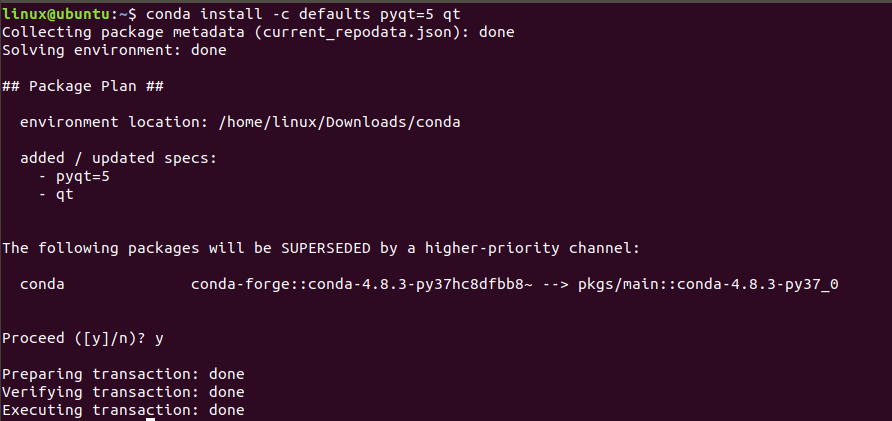
그렇다면 이제부터 linux에 orange를 설치해보자



Conda config –add channels conda-forge

Conda install orange3

이 과정에서 첫번째 code에서 오타 때문에 오류가 났다. 큰 오류는 아니고 remove option을 통해 지워주면 된다.



이를 통해 설치를 완료했다.

다음으로 scOrange를 설치하기 위해 다음과 같은 과정을 따랐다.

| code

Mkdir scorange

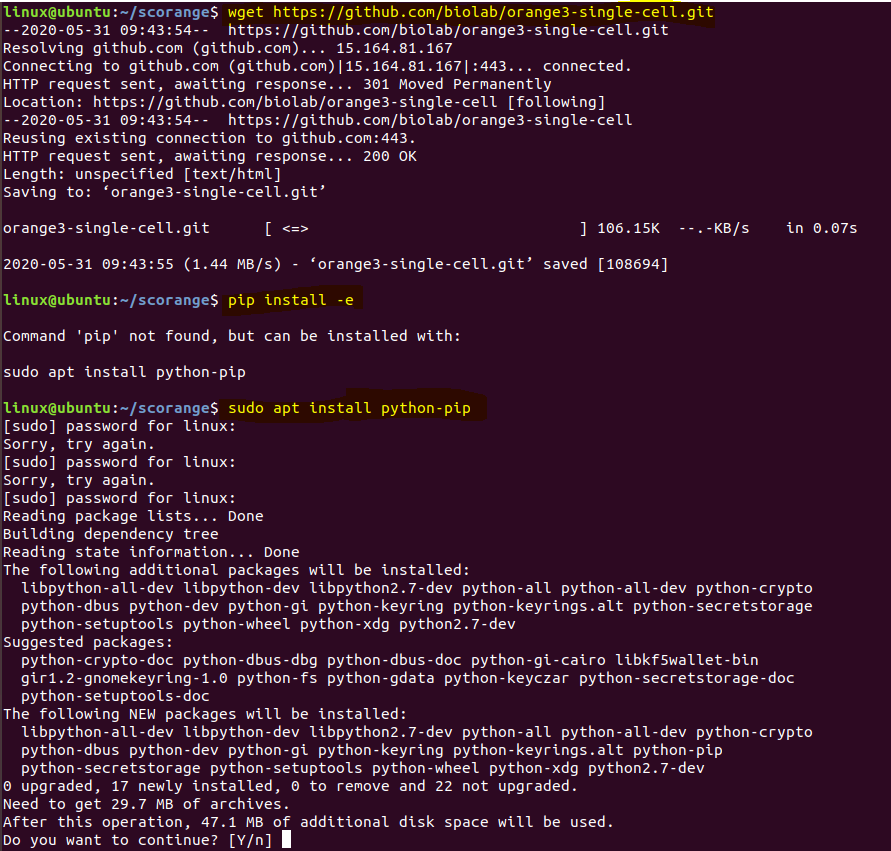
Git clone https://github.com/biolab/orange3-single-cell

Pip install .

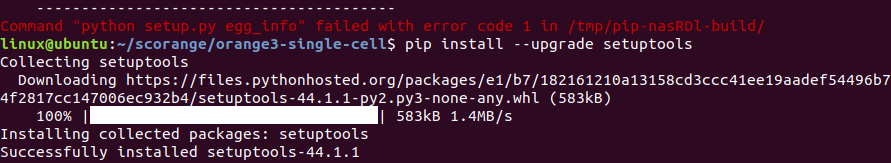
Pip을 설치하지 않았는데 이 사실을 몰랐었다.

Sudo apt install python-pip

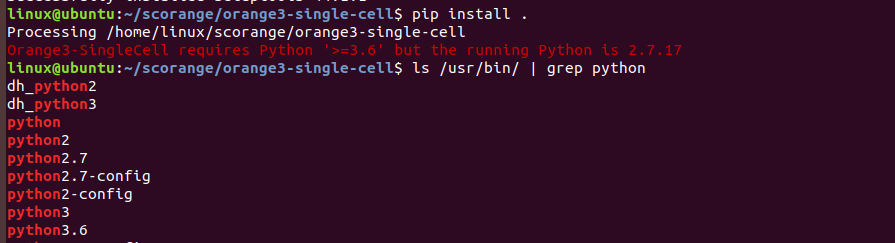
을 통해 pip을 성공적으로 설치하였다.

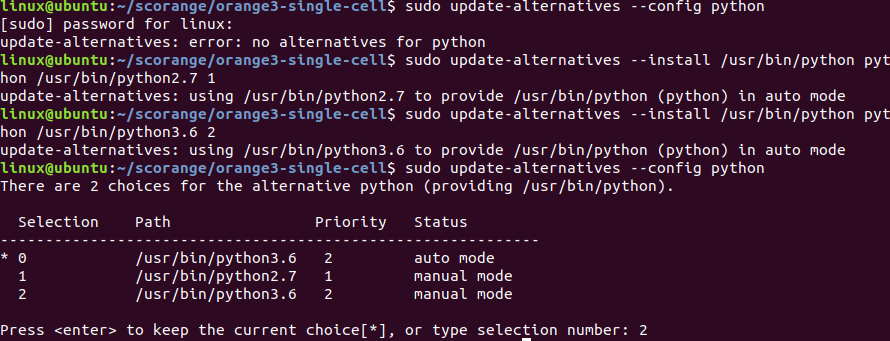


Pip install . 을 하자 다음과 같은 오류가 발생하였다. 그래서 setuptools를 upgrade 해주었다.

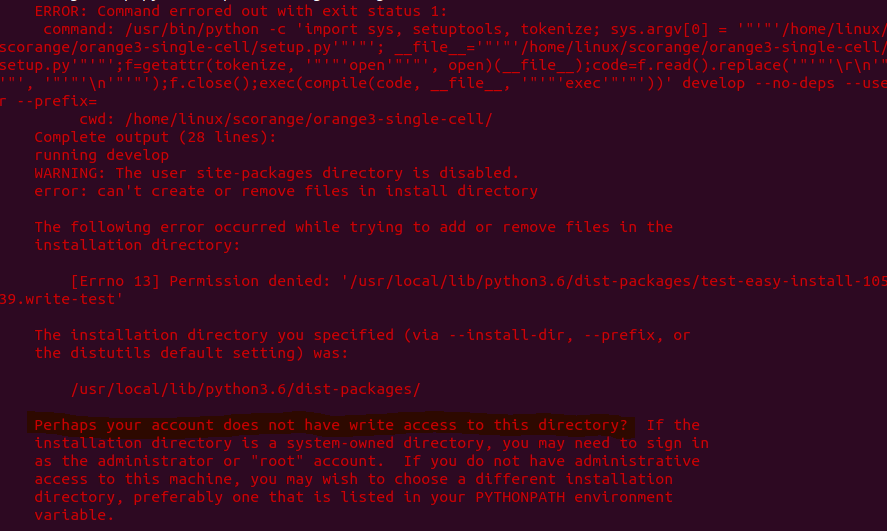


그러나 이번에는 python 버전이 맞지 않다는 오류가 발생하였다. 이에 버전을 바꾸기 위해 update-alternatives --install [symbolic link path] python [real path] number 이 명령어를 사용하여 버전을 3.6으로 바꿔주었다.





이다음에 pip installl -e . 을 했는데 directory에 권한이 없어서 오류가 발생했다.



Sudo 명령을 앞에 붙여 줌으로써 해결했다.

이제 실행만 하면 되는데 PyQt5를 깔아주지 않아 실행이 되지 않았다. 이에 이를 설치해주었다. 그런데 그럼에도 계속해서 오류가 발생하였고 다양한 방법으로 오류를 잡아 보려 했으나 실패하였다. 급기야 메모리가 부족한 문제까지 발생하게 되어 scOrange를 설치하는 것은 다음에 다시 시도해 보기로 하였다.

Week12 – lnav 설치

이번주는 이전과는 조금 다른 것을 설치해보려고 한다.

나는 이 수업 외에 데이터센터프로그래밍이라는 수업을 듣는다. 이 수업에는 docker나 쿠버네테스를 활용하여 log정보를 확인하는 것들이 종종 등장한다.

검색을 하던 중 이러한 log정보를 좀 더 보기 쉽게 만드는 tool이 있다는 것을 알게 되었다.

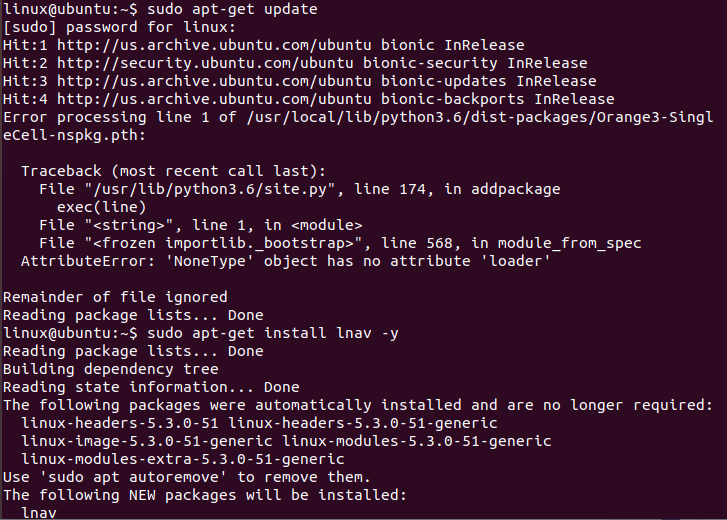
그래서 오늘은 lnav를 설치해볼 것이다.

| code

Sudo apt-get install lnav -y

Lnav -V

\*\*Ubuntu 16.04 default repository에 lnav가 있기 때문에 손쉽게 다운 받을 수 있다.



두번째 명령을 통해 lnav의 버전을 확인해보자

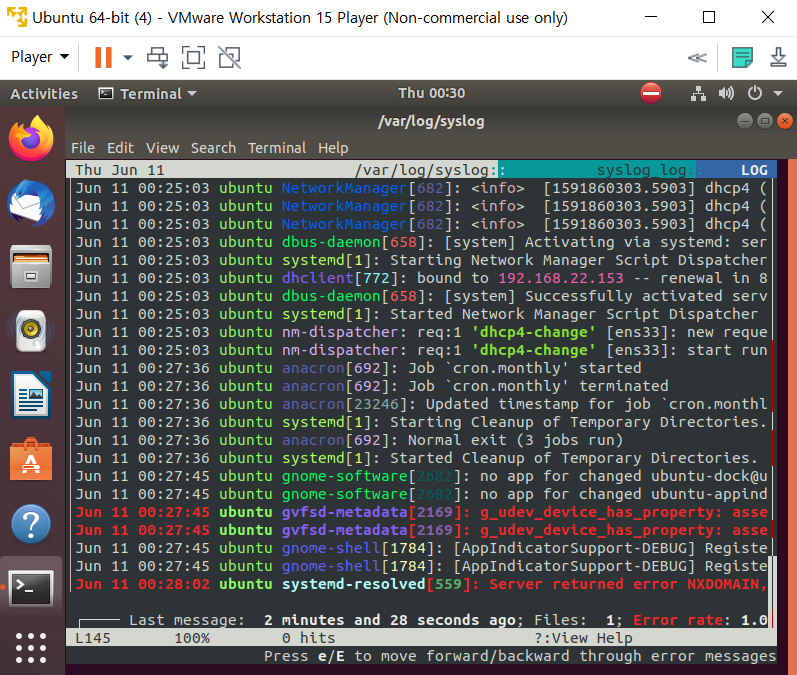


잘 나오는 것으로 보아 성공적으로 설치한 것을 알 수 있다.

이제 이것을 직접 실행시켜보자

| code

Sudo lnav



위와 같은 화면이 잘 뜨는 것을 확인할 수 있었다.

Week13-devops

도커에서 jenkins를 사용한 적이 있는데 linux에도 설치해서 이용해보면 좋은 경험이 될 것 같아 설치해 보기로 하였다.

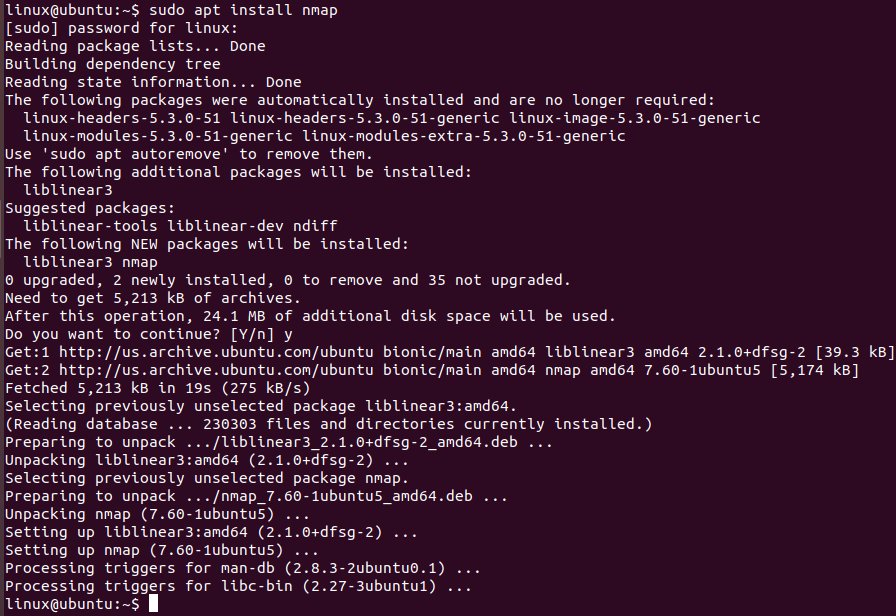
Jenkins는 devops를 실현시키기 위한 일종의 tool이다

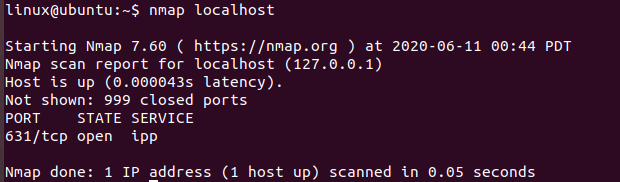
설치하기에 앞서 ubuntu에서 사용되는 포트번호를 확인하자. 그러기 위해 우선 nmap을 깔아준다.

| code

Sudo apt install nmap

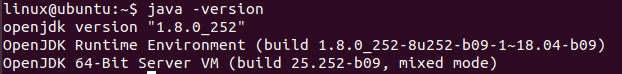
Nmap localhost





사실 사용하는 게 없다. 그런데 혹시나 jenkins의 포트번호를 잘 못 설정할 수 있으니 꼭 확인하도록 하자 (Jenkins의 포트번호는 9711로 하자)

Jenkins를 설치하는데는 java가 필요하므로 설치된 자바가 있는 지 확인한다.



Jenkins 설치를 위한 repository key를 추가한다

| code

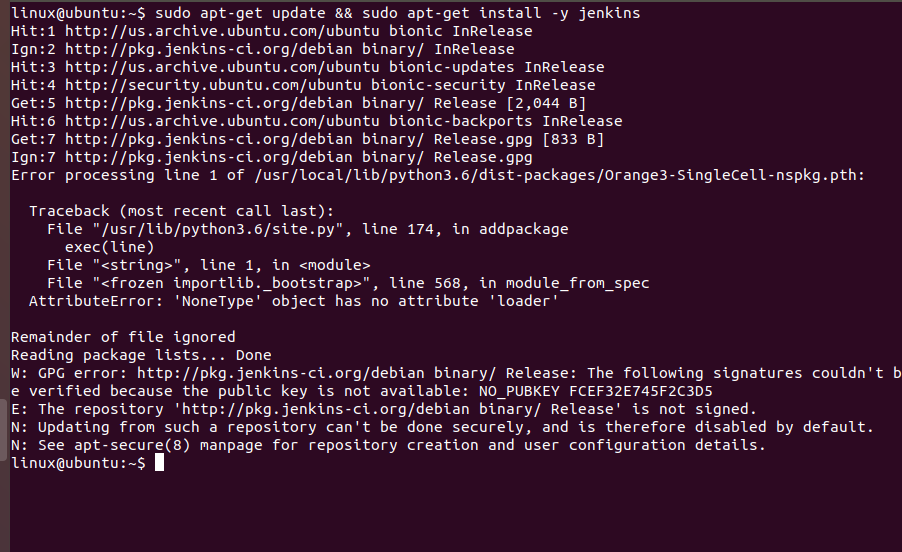
Sudo wget -q -O – <https://pkg.jenkins-ci.io/debian/jenkins-ci.org.key> | sudo apt-key add –



처음에 약간 오타가 있어서 실수가 있었지만 성공적으로 추가해주었다.

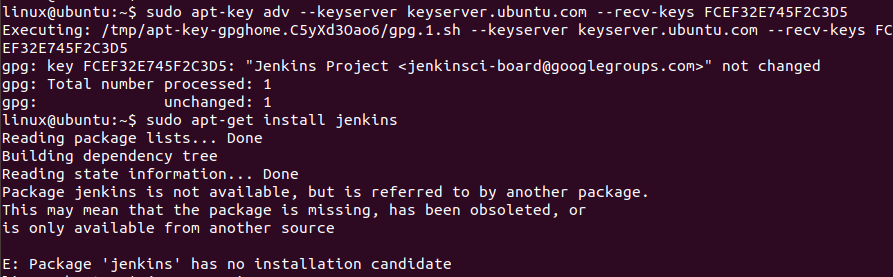
이번에는 repository를 추가한다.

이후에 업데이트를 한 후 jenkins를 설치한다.



그랬는데 이런 오류가 발생했다. 이것은 키가 틀렸다는 것이다.

그래서 직접 홈페이지로 들어가 FCEF32E745F2C3D5에 맞는 키를 찾은 후 등록해주자



그런데도 위와 같은 오류가 났다. 찾아보니 ubuntu에서 jenkins는 공식저장소에서 사라졌다고 한다. 그래서 short version을 써야 한다고 한다. 아래와 같은 명령을 통해 잘 설치할 수 있었다.

