Data Science Edge Node User Guide

Version 0.1

Contents

R Studio DS edge node

Data Science Edge Node is a special machine provided by BDPaaS, which offers more computing power, more administrative privileges and cutting edge tools like H2O, Anaconda, Jupyter Zepplin, R server and many more. On top of it, it provides flexibility to add and install more libraries/packages when required.

R on data science edge node works in a client-server architecture, where the R code is written in a browser based GUI studio, the actual code runs in R server in the edge node's powerful machine.

Prerequisites 1 – Create user directory

Before you can proceed further, make sure that your user directory is setup in Bigdata users home path, check if the following directory exists or not? If it doesn't exist, then create one.

/mapr/datalake/other/aes_ucee_bd_pr_2/users/**yourmsid**

Where yourmsid needs to be replaced with your actual msid value in the path.

Prerequisite 2 - Setup .profile file

Similar to shared edge nodes, you need to set up your .profile file in your home directory first.

To do that, open putty for host apsrp09132 and port 22, go to your home directory by typing

To edit .profile file

1. Open putty for host **apsrp09132** and port 22,

2. Then goto your home directory by entering the following command
   cd /home/**yourmsid**
   (replace the placeholder "yourmsid" with your actual one)
3. Open the .profile file in vi editor, by entering the following command in putty:

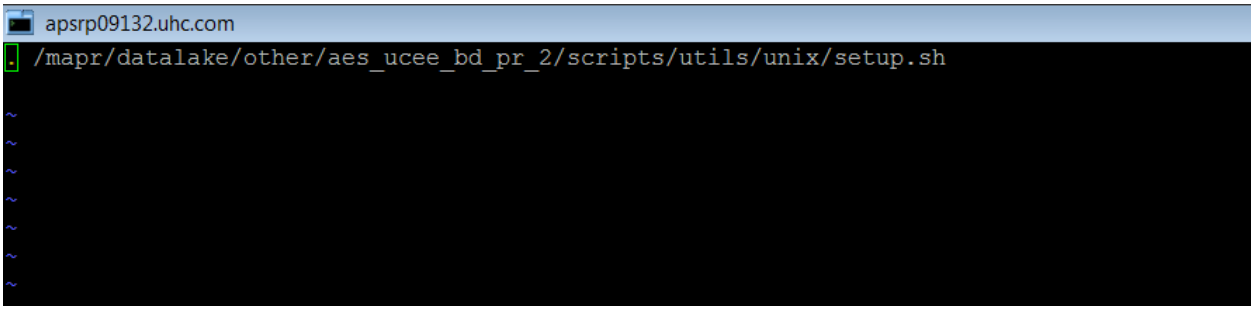
   vi .profile

   Then press '**I**', and it will show insert in the left hand bottom side of the vi interface.

   Now type the following in .profile file

   . /mapr/datalake/other/aes_ucee_bd_pr_2/scripts/utils/unix/setup.sh

   (Note the whitespace between '.' and '/')



4. To save the .profile file press Esc, cursor comes down to the bottom left corner and type the following command and hit enter.

   :wq

5. After this, close the putty session and login again.

Setup RStudio

Once you have setup .profile file as per the step above, execute the rstudio setup tool by entering the following command

sh setup_dsen_rstudio.sh

```
$ sh setup_dsen_rstudio.sh
Configuring .Renviron
RStuio setup was completed successfully
akunal@apsrp09132:/home/akunal
$
```

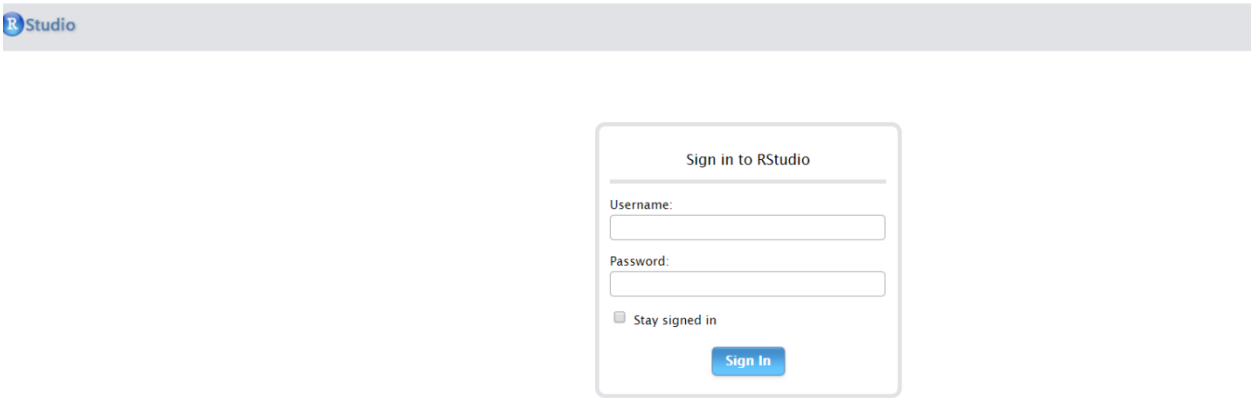
Check the output logs, and verify if your configuration was completed successfully. In case, it says FAILED, contact Data engineering team for the further assistance. Data engineering team can be reached out @ aes_bdpaas_intake@optum.com

RStudio browser application

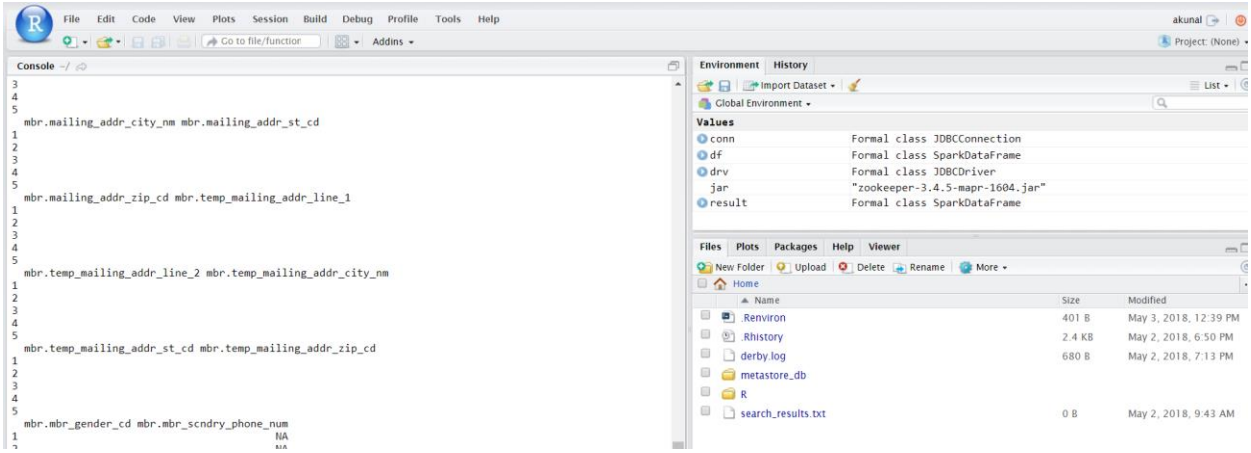
To use R Studio in DS edge node, hit the following URL

http://apsrp09132:8787

This will throw the login popup, enter your MS credentials



On successful login, you will be able to see the below screen



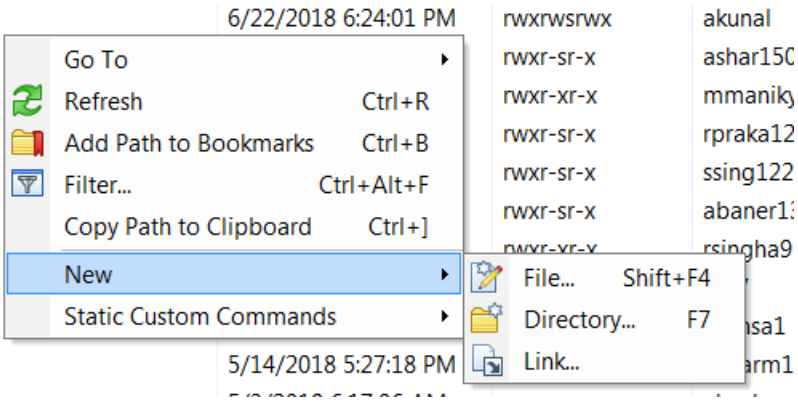
Reading a file

R Studio Server can load and read files from BDPaaS volume path, which means, files stored on Hadoop can be directly accessed by R Studio. What this also means is, your regular files, which you may have stored from different sources to BDPaaS volume path can be accessed. Please remember, DS edge node is using the same development cluster, and hence **any** files stored on BDPaaS volume path (irrespective of environment, e.g. from shared edge node dbsld0068 or DS edge node apsrd00633) can be accessed.

As per the existing practice, user's specific files can be stored in the following volume path

/mapr/datalake/other/aes_ucee_bd_pr_2/users/yourmsid

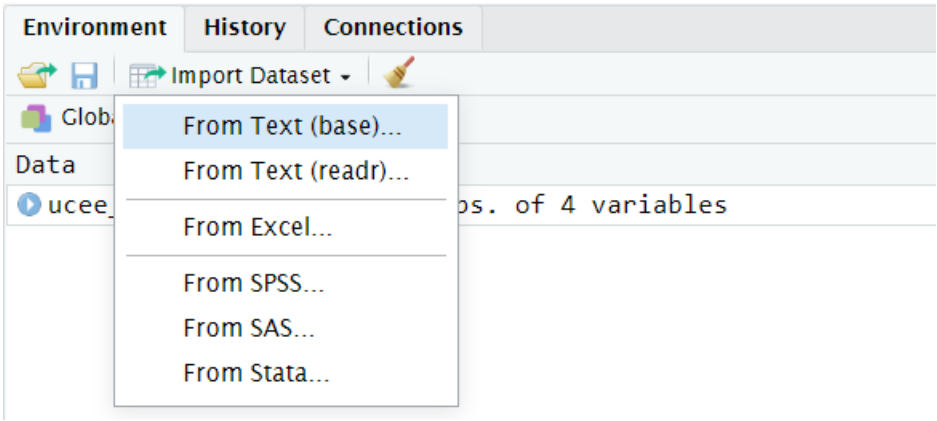
If you don't see a directory for you in the users directory, you may create one yourself with Winscp. Your user directory should already exist by the means of completing prerequisite 1.



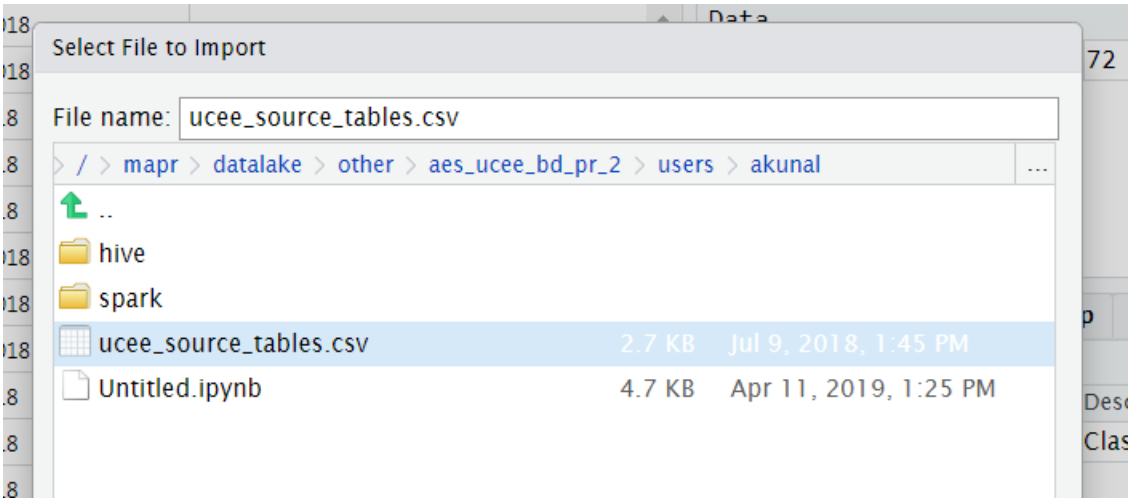
With your user directory available inside users directory, you can transfer files here to be used in R.

There are 2 ways to read files into DS node R studio.

1. Use “Import Dataset” option from the UI. Please refer the screenshot below



In the opened dialog window , enter the fully qualified path of the file



And then proceed by clicking import button.

ucee_source_tables

Filter

	DB_NAME	TBL_NAME	LOAD_DATE	LAST_ACCESS_TIME
1	UCEE	ALL_APPS_2015	5-Mar-18	4/12/2018
2	UCEE	ALL_APPS_2016	5-Mar-18	4/12/2018
3	UCEE	ALL_APPS_2017	6-Mar-18	7/3/2018
4	UCEE	ALL_APPS_2018	5-Jul-18	7/6/2018
5	UCEE	AMLK	23-Mar-18	7/5/2018
6	UCEE	APPS_2017	16-Jan-18	4/24/2018
7	UCEE	APPS_2018NEW	6-Mar-18	4/24/2018
8	UCEE	AUX_REP	24-May-18	6/30/2018
9	UCEE	BRAND	6-Jul-18	7/6/2018
10	UCEE	CAMP	6-Jul-18	7/6/2018
11	UCEE	CAMP_CELL	6-Nov-17	7/5/2018

2. You can directly use following code to import data from a file
   base_data <-
   read.csv("/mapr/datalake/other/aes_ucee_bd_pr_2/users/akunal/ucee_source_tables.csv", header=TRUE, sep =",")

   ** with the above code, you can also read other delimited files, you just need to change the sep option.

Connecting from R to Hive

There are 2 steps required

1. Create your logininfo.R file in your BDPaaS user directory. If your msid is bob1, then your BDPaaS user directory is
   /mapr/datalake/other/aes_ucee_bd_pr_2/users/users/**bob1**

   Open a putty session to apsrp09132, and go to your BDPaaS user directory
   cd /mapr/datalake/other/aes_ucee_bd_pr_2/users/users/**bob1**

   Then create logininfo.R file
   vi logininfo.R, and then paste following code

```
username <- "yourusername"
password <- "yourpassword"
```

If you prefer, you can also create logininfo.R in your local machine, and then transfer the file to your home directory via Winscp.

Once the file has been created, go to your home directory

cd /mapr/datalake/other/aes_ucee_bd_pr_2/users/users/**bob1**

and then execute the following command
chmod 700 logininfo.R

2. To connect from R to Hive, use the following code

```
library("rJava")
library("RJDBC")
source("/mapr/datalake/other/aes_ucee_bd_pr_2/users/akunal/R/logininfo.R")
drv <- JDBC("org.apache.hive.jdbc.HiveDriver", "/opt/mapr/hive/hive-2.1/lib/hive-jdbc-2.1.1-mapr-1803.jar")
for(jar in list.files('/opt/mapr/hive/hive-2.1/lib/')){ .jaddClassPath(paste("/opt/mapr/hive/hive-2.1/lib/",jar,sep=""))}
for(jar in list.files('/opt/mapr/lib')){ .jaddClassPath(paste("/opt/mapr/lib/",jar,sep=""))}
conn <- dbConnect(drv, "jdbc:hive2://dbslp0569:10844/default", username, password)


# Actual query below

some_5_members <- dbGetQuery(conn, "select src_sys_srcid, mbr_id_cd from raw_ucee_nr.mbr limit 5")
```

Spark from RStudio

To connect from R to Spark, you can use the following code.

Prerequisite:

You need to have a valid maprlogin session on the DS edge node. If you have not done maprlogin password on apsrd00633, then open a putty session for apsrd00633, and complete maprlogin flow.

SparkR code

Please note that, lib.loc from the below statement cannot be omitted.

```
library(SparkR, lib.loc = "/opt/mapr/spark/spark-2.2.1/R/lib")
sparkR.session(enableHiveSupport = TRUE)
result <- sql("select count(*) from raw_ucee_nr.MBR")

# The dataframe result will contain one row and one column, which is the count value
collect(result)
```

Available Packages

As part of DS edge node administration, data engineering team reviews and installs package and libraries required. With the help of consultations held with R team members, DE team has installed most widely used R packages. Given below is the list of available packages

Package	Version
abind	1.4-5
acepack	1.4.1
ada	2.0-5
amap	0.8-16
arules	1.6-3
askpass	1.1
backports	1.1.3
base64enc	0.1-3
BH	1.69.0-1
bigmemory	4.5.33
bigmemory.sri	0.1.3
bit	1.1-14
bit64	0.9-7
bitops	1.0-6
blob	1.1.1
BradleyTerry2	1.0-9
brew	1.0-6
brglm	0.6.2
callr	3.2.0
car	3.0-2
carData	3.0-2
caret	6.0-82

caTools	1.17.1.2
cba	0.2-20
cellranger	1.1.0
checkmate	1.9.1
chron	2.3-53
clipr	0.5.0
clisymbols	1.2.0
commonmark	1.7
curl	3.3
data.table	1.12.2
DBI	1.0.0
desc	1.2.0
devtools	2.0.2
dichromat	2.0-0
dplyr	0.8.0.1
e1071	1.7-1
ellipsis	0.1.0
evaluate	0.13
forcats	0.4.0
foreach	1.4.4
formatR	1.6
Formula	1.2-3
fs	1.2.7
gdata	2.18.0
generics	0.0.2
geosphere	1.5-7
ggplot2	3.1.0
gh	1.0.1
git2r	0.25.2
glmnet	2.0-16
googleVis	0.6.3
gower	0.2.0
gplots	3.0.1.1
gridExtra	2.3
gsubfn	0.7
gtools	3.8.1
h2o	3.22.1.1
haven	2.1.0
highr	0.8
Hmisc	4.2-0
hms	0.4.2
htmlTable	1.13.1
htmltools	0.3.6

htmlwidgets	1.3
httpuv	1.5.1
httr	1.4.0
ini	0.3.1
ipred	0.9-8
iterators	1.0.10
itertools	0.1-3
jsonlite	1.6
knitr	1.22
later	0.8.0
latticeExtra	0.6-28
lava	1.6.5
lda	1.4.2
LDAvis	0.3.2
lme4	1.1-21
lubridate	1.7.4
magrittr	1.5
mapproj	1.2.6
maps	3.3.0
maptools	0.9-5
markdown	0.9
MatrixModels	0.4-1
memoise	1.1.0
mime	0.6
minqa	1.2.4
missForest	1.4
mlbench	2.1-1
ModelMetrics	1.2.2
modeltools	0.2-22
nloptr	1.2.1
NLP	0.2-0
numDeriv	2016.8-1
openssl	1.3
openxlsx	4.1.0
outliers	0.14
pbkrtest	0.4-7
pkgbuild	1.0.3
pkgload	1.0.2
plogr	0.2.0
plyr	1.8.4
pmmlTransformations	1.3.2
png	0.1-7
prettyunits	1.0.2

processx	3.3.0
prodlim	2018.04.18
profileModel	0.6.0
progress	1.2.0
promises	1.0.1
proto	1.0.0
proxy	0.4-23
ps	1.3.0
purrr	0.3.2
quantreg	5.38
qvcalc	0.9-1
randomForest	4.6-14
randomForestSRC	2.8.0
Rankcluster	0.94
rcmdcheck	1.3.2
RColorBrewer	1.1-2
Rcpp	1.0.1
RcppEigen	0.3.3.5.0
RcppRoll	0.3.0
RCurl	1.95-4.12
readr	1.3.1
readxl	1.3.1
recipes	0.1.5
rematch	1.0.1
remotes	2.0.3
reshape2	1.4.3
RgoogleMaps	1.4.3
rio	0.5.16
rJava	0.9-11
RJDBC	0.2-7.1
rjson	0.2.20
RJSONIO	1.3-1.1
ROAuth	0.9.6
ROCR	1.0-7
rpart.plot	3.0.6
rprojroot	1.3-2
RSQLite	2.1.1
rstudioapi	0.1
RWekajars	3.9.3-1
sessioninfo	1.1.1
shiny	1.3.0
slam	0.1-45
SnowballC	0.6.0

sourcetools	0.1.7
sp	1.3-1
SparseM	1.77
sqldf	0.4-11
SQUAREM	2017.10-1
statmod	1.4.30
stringi	1.4.3
stringr	1.4.0
sys	3.1
tidyr	0.8.3
tidyselect	0.2.5
timeDate	3043.102
twitteR	1.1.9
usethis	1.5.0
viridis	0.5.1
whisker	0.3-2
wordcloud	2.6
xfun	0.6
xopen	1.0.0
xtable	1.8-3
yaml	2.2.0
zip	2.0.1

New Package Installation Process

You can drop a mail to data engineering team to install a missing package. Data engineering team does the following, as part of installation process

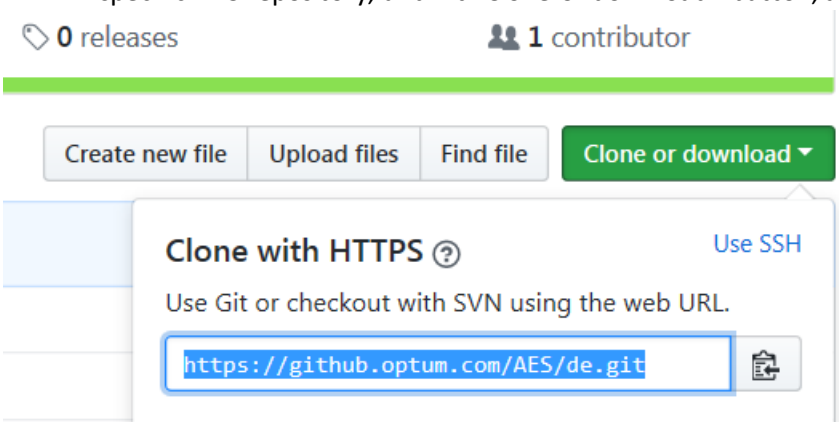
1. Assess different aspects of the library, e.g. no of releases, stability, security and committer's responsiveness
2. Install OS level dependencies, and change configurations, involve different teams, e.g. BDPaaS platform team, Unix Server Management team if required
3. Install the library, and do primary validations
4. Make the library available for all users to use

Using source version control R to Github

It's very important to use a source version control system to better manage projects, which involve writing code regularly. Not only users will be able to store their code in a very organized fashion, also, a collaborative work environment will be promoted, and so will culture.

In AES, we are using GIT as the source version control software. To use GIT in R, you can follow these steps

1. First, select the AES repository, from which you want to create the R project. For that, goto the specific AES repository, and hit "Clone or download" button, as shown in the screenshot below

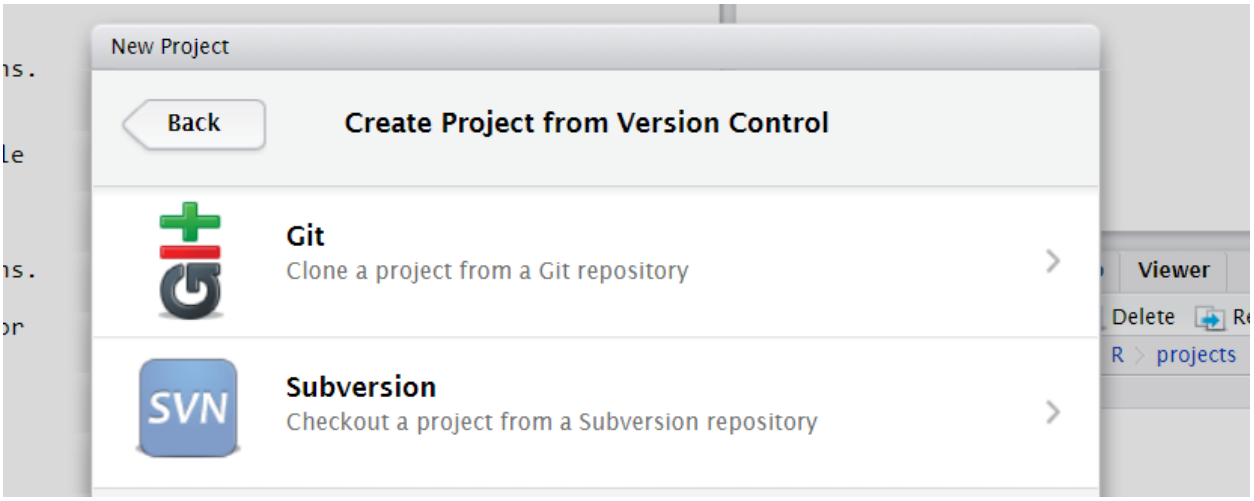


2. Now you will need to edit the repository URL to include your github credentials (MS credentials). Edit the URL to have that in the following format

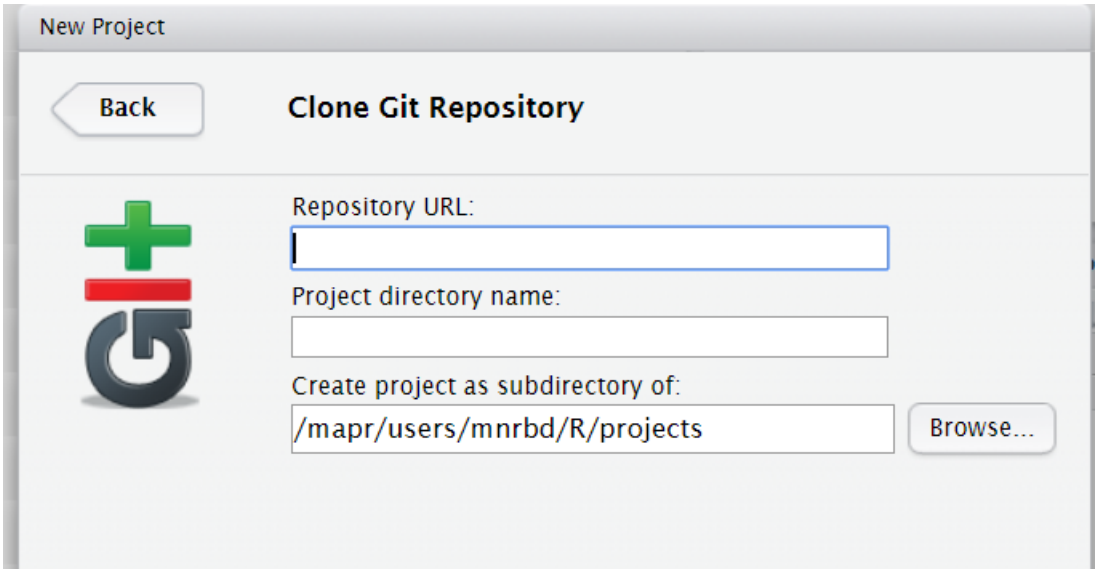
   https://msid:mspassword@github.optum.com/AES/de.git

3. Now you are ready to check out the project as specific Rstudio Git project, to do that open DSEN rstudio by going to http://apsrp09132:8787, and then

File -> new project -> version control -> Git



Select Git in the screen above, and then enter the details accordingly



Jupyter

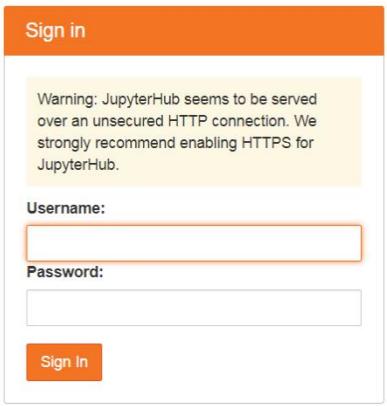
Please make sure that you have completed prerequisites mentioned in the beginning of this document before going ahead.

1. Do a maprlogin first on apsrp09132, if not done for the day.

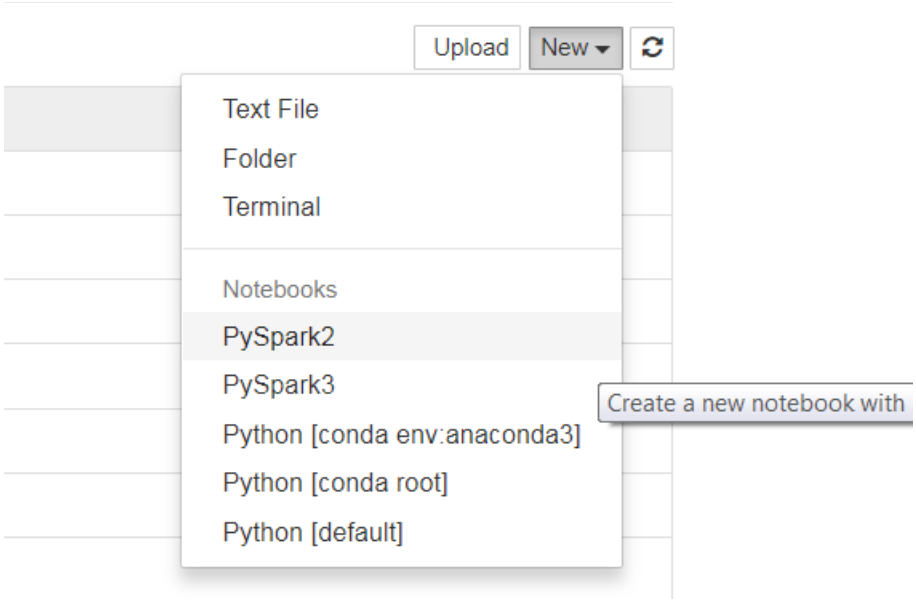
2. Jupyter is available at http://apsrp09132:8000 on DS edge node

   On hitting the URL, login popup will come, enter your msid and ms password

jupyter



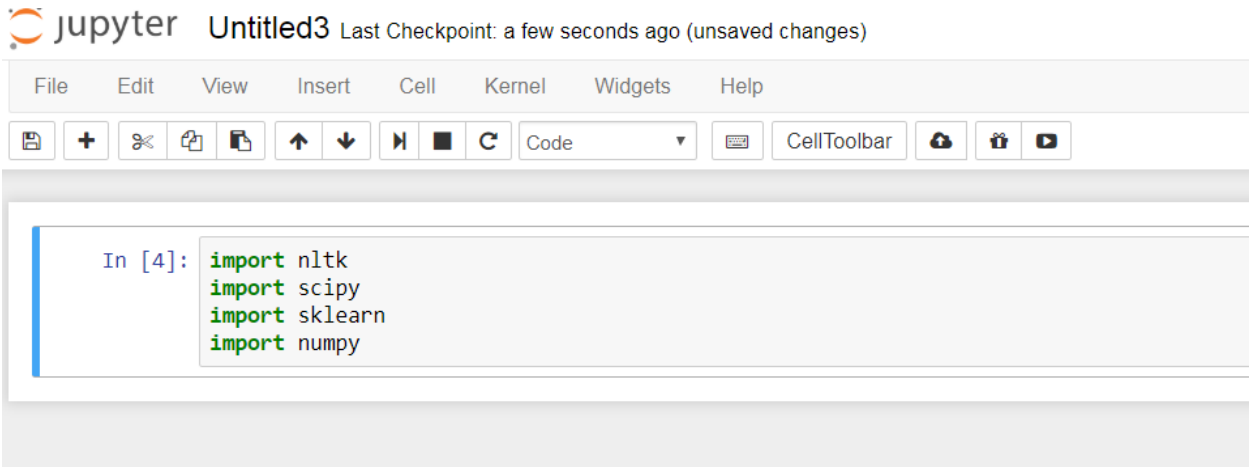
After successful login , you will be able to see the Jupyter home page, and then you can create a python anaconda notebook by selecting the highlighted menu in the screenshot given below



.

Available python libraries

There are many python libraries available out of the box, for different data science activities, text mining, seintific computations, model building.



Package	Version
alabaster	0.7.10
anaconda-clean	1
anaconda-client	1.5.1
anaconda-navigator	1.3.1
argcomplete	1.0.0
asn1crypto	0.22.0
astroid	1.4.7
astropy	1.2.1
attrs	15.2.0
automat	0.0.0
babel	2.3.4
backports-abc	0.4
backports.shutil-get-terminal-size	1.0.0
backports.ssl-match-hostname	3.4.0.2
beautifulsoup4	4.5.1
bitarray	0.8.1
blaze	0.10.1
bokeh	0.12.2
boto	2.48.0
bottleneck	1.1.0

bz2file	0.98
cdecimal	2.3
certifi	2016.2.28
cffi	1.10.0
chest	0.2.3
click	6.6
cloudpickle	0.2.1
clyent	1.2.2
colorama	0.3.7
conda-build	2.0.2
conda	4.2.9
configobj	5.0.6
configparser	3.5.0
constantly	15.1.0
contextlib2	0.5.3
cryptography	1.8.1
cssselect	1.0.1
cycler	0.10.0
cymem	1.31.2
cython	0.24.1
cytoolz	0.8.2
dask	0.11.0
datashape	0.5.2
decorator	4.1.2
dill	0.2.6
docutils	0.12
dynd	0.7.3.dev1
enum34	1.1.6
et-xmlfile	1.0.1
fastcache	1.0.2
filelock	2.0.6
flask-cors	2.1.2
flask	0.11.1
ftfy	4.4.3
funcsigs	1.0.2
functools32	3.2.3.post2
futures	3.0.5
gensim	2.3.0
gevent	1.1.2
greenlet	0.4.10
grin	1.2.1
h5py	2.6.0
heapdict	1.0.0

html5lib	1
hyperlink	17.1.1
idna	2.6
imagesize	0.7.1
incremental	16.10.1
ipaddress	1.0.18
ipykernel	4.5.0
ipython-genutils	0.2.0
ipython	5.1.0
ipywidgets	5.2.2
itsdangerous	0.24
jdcal	1.2
jedi	0.9.0
jinja2	2.8
jsonschema	2.6.0
jupyter-client	4.4.0
jupyter-console	5.0.0
jupyter-core	4.3.0
jupyter	1.0.0
lazy-object-proxy	1.2.1
llvmlite	0.13.0
locket	0.2.0
lxml	3.8.0
markdown	2.6.9
markupsafe	0.23
matplotlib	1.5.3
mistune	0.7.3
mpmath	0.19
msgpack-python	0.2.3
multipledispatch	0.4.8
murmurhash	0.26.4
nb-anacondacloud	1.2.0
nb-conda-kernels	2.0.0
nb-conda	2.0.0
nbconvert	4.2.0
nbformat	4.4.0
nbpresent	3.0.2
networkx	1.11
nltk	3.2.1
nose	1.3.7
notebook	4.2.3
numba	0.28.1+0.gfe99fbc.dirty
numexpr	2.6.1

numpy	1.13.1
odo	0.5.0
openpyxl	2.3.2
packaging	16.8
pandas	0.18.1
parsel	1.2.0
partd	0.3.6
path.py	0.0.0
pathlib2	2.1.0
pathlib	1.0.1
patsy	0.4.1
pep8	1.7.0
pexpect	4.0.1
pickleshare	0.7.4
pillow	3.3.1
pip	9.0.1
pkginfo	1.3.2
plac	0.9.6
plotly	2.0.11
ply	3.9
preshed	1.0.0
prompt-toolkit	1.0.3
psutil	4.3.1
ptyprocess	0.5.1
py	1.4.31
pyasn1-modules	0.0.8
pyasn1	0.2.3
pycairo	1.10.0
pycosat	0.6.1
pycparser	2.18
pycrypto	2.6.1
pycurl	7.43.0
pydispatcher	2.0.5
pyflakes	1.3.0
pygments	2.1.3
pylint	1.5.4
pyopenssl	17.0.0
pyparsing	2.2.0
pytest	2.9.2
python-dateutil	2.5.3
pytz	2017.2
pyyaml	3.12
pyzmq	15.4.0

qtawesome	0.3.3
qtconsole	4.2.1
qtpy	1.1.2
queuelib	1.4.2
redis	2.10.5
regex	2017.4.5
requests	2.14.2
rope	0.9.4
uruamel-yaml	#NAME?
scikit-image	0.12.3
scikit-learn	0.19.0
scipy	0.19.1
scrapy	1.3.3
service-identity	17.0.0
setuptools	36.4.0
simplegeneric	0.8.1
singledispatch	3.4.0.3
six	1.10.0
smart-open	1.5.3
snowballstemmer	1.2.1
sockjs-tornado	1.0.3
spacy	1.8.2
sphinx	1.4.6
spyder	3.0.0
sqlalchemy	1.0.13
statsmodels	0.6.1
sympy	1
tables	3.2.3.1
termcolor	1.1.0
terminado	0.6
thinc	6.5.2
toolz	0.8.2
tornado	4.4.1
tqdm	4.15.0
traitlets	4.3.2
twisted	17.5.0
ujson	1.35
unicodecsv	0.14.1
w3lib	1.17.0
wcwidth	0.1.7
werkzeug	0.11.11
wheel	0.29.0
widgetsnbextension	1.2.6

wrapt	1.10.11
xlrd	1.0.0
xlsxwriter	0.9.3
xlwt	1.1.2
zope.interface	4.4.2

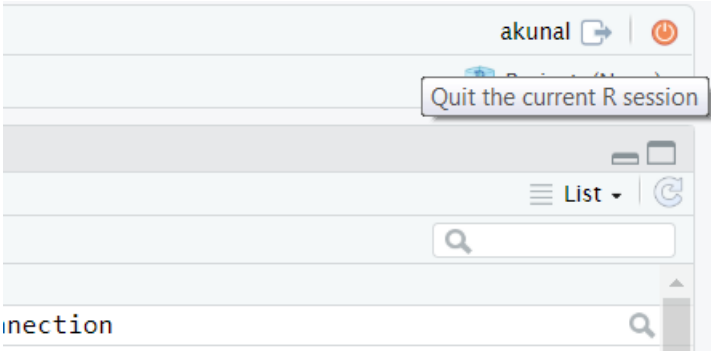
DSEN Don'ts

Except .profile file, don't create any file in your unix home location e.g. /home/yourmsid

Do not keep the rstudio or Jupyter running if you are not using it. Follow the steps given below to close the sessions when you are done.

To close Jupyter session and notebooks, go to the files tab of the parent most window/orginal window, and then select all the running notebooks, and then click on the shutdown.

To close Rstudio session, click the red power button, and then chose if you would want to save the current session data or not in the next popup.



Just close the browser windo/tab, when you see this