UCSCXenaTools: Download Public Cancer Genomic Data from UCSC Xena Hubs

Shixiang Wang

2019-07-23

UCSCXenaTools is an R package for downloading and exploring data from **UCSC Xena data hubs**, which are

> a collection of UCSC-hosted public databases such as TCGA, ICGC, TARGET, GTEx, CCLE, and others. Databases are normalized so they can be combined, linked, filtered, explored and downloaded.
>
> – UCSC Xena

If you use this package in academic field, please cite:

```
Wang, Shixiang, et al. "The predictive power of tumor mutational burden
    in lung cancer immunotherapy response is influenced by patients' sex."
    International journal of cancer (2019).
```

Installation

Install stable release from CRAN with:

```
install.packages("UCSCXenaTools")
```

You can also install devel version of **UCSCXenaTools** from github with:

```
# install.packages('remotes')
remotes::install_github("ShixiangWang/UCSCXenaTools")
```

Data Hub List

All datasets are available at https://xenabrowser.net/datapages/.
Currently, **UCSCXenaTools** supports 10 data hubs of UCSC Xena.

- UCSC Public Hub: https://ucscpublic.xenahubs.net
- TCGA Hub: https://tcga.xenahubs.net
- GDC Xena Hub: https://gdc.xenahubs.net
- ICGC Xena Hub: https://icgc.xenahubs.net
- Pan-Cancer Atlas Hub: https://pancanatlas.xenahubs.net
- GA4GH (TOIL) Hub: https://toil.xenahubs.net
- Treehouse Hub: https://xena.treehouse.gi.ucsc.edu
- PCAWG Hub: https://pcawg.xenahubs.net
- ATAC-seq Hub: https://atacseq.xenahubs.net

- Singel Cell Xena hub: https://singlecell.xenahubs.net

If any url of data hubs are changed or a new data hub is online, please remind me by emailing to w_shixiang@163.com or opening an issue on GitHub.

Usage

Download UCSC Xena Datasets and load them into R by **UCSCXenaTools** is a workflow with `generate`, `filter`, `query`, `download` and `prepare` 5 steps, which are implemented as `XenaGenerate`, `XenaFilter`, `XenaQuery`, `XenaDownload` and `XenaPrepare` functions, respectively. They are very clear and easy to use and combine with other packages like `dplyr`.

To show the basic usage of **UCSCXenaTools**, we will download clinical data of LUNG, LUAD, LUSC from TCGA (hg19 version) data hub.

XenaData data.frame

Begin from version `0.2.0`, **UCSCXenaTools** uses a `data.frame` object (built in package, someone may call it `tibble`) `XenaData` to generate an instance of `XenaHub` class to record general information of all datasets of UCSC Xena Data Hubs.

You can load `XenaData` after loading `UCSCXenaTools` into R.

```
library(UCSCXenaTools)
#> ========================================================================
#> UCSCXenaTools version 1.2.4
#> Github page: https://github.com/ShixiangWang/UCSCXenaTools
#> Documentation: https://shixiangwang.github.io/UCSCXenaTools/
#>
#> If you use it in published research, please cite:
#> Wang, Shixiang, et al. "The predictive power of tumor mutational burden
#>     in lung cancer immunotherapy response is influenced by patients' sex."
#>     International journal of cancer (2019).
#> ========================================================================
#>
data(XenaData)

head(XenaData)
#> # A tibble: 6 x 17
#>   XenaHosts XenaHostNames XenaCohorts
#>   <chr>     <chr>         <chr>
#> 1 https://~ publicHub     Acute lymp~
#> 2 https://~ publicHub     Acute lymp~
```

```
#> 3 https://~ publicHub     Acute lymp~
#> 4 https://~ publicHub     Breast Can~
#> 5 https://~ publicHub     Breast Can~
#> 6 https://~ publicHub     Breast Can~
#> # ... with 14 more variables:
#> #   XenaDatasets <chr>, SampleCount <chr>,
#> #   DataSubtype <chr>, Label <chr>,
#> #   Type <chr>, AnatomicalOrigin <chr>,
#> #   SampleType <chr>, Tags <chr>,
#> #   ProbeMap <chr>, LongTitle <chr>,
#> #   Citation <chr>, Version <chr>,
#> #   Unit <chr>, Platform <chr>
```

Names of all hub names/urls can be accessed by object `.xena_hosts`:

```
UCSCXenaTools:::.xena_hosts
#>    https://ucscpublic.xenahubs.net
#>                          "publicHub"
#>          https://tcga.xenahubs.net
#>                            "tcgaHub"
#>           https://gdc.xenahubs.net
#>                            "gdcHub"
#>          https://icgc.xenahubs.net
#>                            "icgcHub"
#>          https://toil.xenahubs.net
#>                            "toilHub"
#>   https://pancanatlas.xenahubs.net
#>                     "pancanAtlasHub"
#> https://xena.treehouse.gi.ucsc.edu
#>                       "treehouseHub"
#>         https://pcawg.xenahubs.net
#>                           "pcawgHub"
#>       https://atacseq.xenahubs.net
#>                         "atacseqHub"
#>    https://singlecell.xenahubs.net
#>                      "singlecellHub"
```

Generate a XenaHub object

This can be implemented by `XenaGenerate` function, which generates `XenaHub` object from `XenaData` data frame.

```
XenaGenerate()
#> class: XenaHub
#> hosts():
#>   https://ucscpublic.xenahubs.net
#>   https://tcga.xenahubs.net
#>   https://gdc.xenahubs.net
#>   https://icgc.xenahubs.net
#>   https://toil.xenahubs.net
#>   https://pancanatlas.xenahubs.net
#>   https://xena.treehouse.gi.ucsc.edu
#>   https://pcawg.xenahubs.net
#>   https://atacseq.xenahubs.net
#>   https://singlecell.xenahubs.net
#> cohorts() (140 total):
#>   Acute lymphoblastic leukemia (Mullighan 2008)
#>   Breast Cancer (Caldas 2007)
#>   Breast Cancer (Chin 2006)
#>   ...
#>   human brain transcriptome (Darmanis PNAS 2015)
#>   mouse cortex and hippocampus (Zeisel Science 2015)
#> datasets() (1646 total):
#>   mullighan2008_public/mullighan2008_500K_genomicMatrix
#>   mullighan2008_public/mullighan2008_public_clinicalMatrix
#>   mullighan2008_public/mullighan2008_SNP6_genomicMatrix
#>   ...
#>   Zeisel/Zeisel_expression_mRNA_log2
#>   Zeisel/Zeisel_expression_phenotype
```

You can set `subset` argument to narrow datasets.

```
XenaGenerate(subset = XenaHostNames == "tcgaHub")
#> class: XenaHub
#> hosts():
#>   https://tcga.xenahubs.net
#> cohorts() (38 total):
#>   TCGA Acute Myeloid Leukemia (LAML)
#>   TCGA Adrenocortical Cancer (ACC)
#>   TCGA Bile Duct Cancer (CHOL)
#>   ...
#>   TCGA Thyroid Cancer (THCA)
#>   TCGA Uterine Carcinosarcoma (UCS)
#> datasets() (879 total):
#>   TCGA.LAML.sampleMap/HumanMethylation27
#>   TCGA.LAML.sampleMap/HumanMethylation450
```

```
#>   TCGA.LAML.sampleMap/Gistic2_CopyNumber_Gistic2_all_data_by_genes
#>   ...
#>   TCGA.UCS.sampleMap/Pathway_Paradigm_RNASeq_And_Copy_Number
#>   TCGA.UCS.sampleMap/mutation_curated_broad
```

You can also use `XenaHub()` to generate a `XenaHub` object for API communication, but it is not recommended.

It's possible to extract info from `XenaHub` object by `hosts()`, `cohorts()` and `datasets()`.

```
xe = XenaGenerate(subset = XenaHostNames == "tcgaHub")
# get hosts
hosts(xe)
#> [1] "https://tcga.xenahubs.net"
# get cohorts
head(cohorts(xe))
#> [1] "TCGA Acute Myeloid Leukemia (LAML)"
#> [2] "TCGA Adrenocortical Cancer (ACC)"
#> [3] "TCGA Bile Duct Cancer (CHOL)"
#> [4] "TCGA Bladder Cancer (BLCA)"
#> [5] "TCGA Breast Cancer (BRCA)"
#> [6] "TCGA Cervical Cancer (CESC)"
# get datasets
head(datasets(xe))
#> [1] "TCGA.LAML.sampleMap/HumanMethylation27"
#> [2] "TCGA.LAML.sampleMap/HumanMethylation450"
#> [3] "TCGA.LAML.sampleMap/Gistic2_CopyNumber_Gistic2_all_data_by_genes"
#> [4] "TCGA.LAML.sampleMap/mutation_wustl_hiseq"
#> [5] "TCGA.LAML.sampleMap/GA"
#> [6] "TCGA.LAML.sampleMap/HiSeqV2_percentile"
```

Pipe operator %>% can also be used here.

```
library(dplyr)
XenaData %>% filter(XenaHostNames == "tcgaHub",
    grepl("BRCA", XenaCohorts), grepl("Path",
        XenaDatasets)) %>% XenaGenerate()
#> class: XenaHub
#> hosts():
#>   https://tcga.xenahubs.net
#> cohorts() (1 total):
#>   TCGA Breast Cancer (BRCA)
#> datasets() (4 total):
#>   TCGA.BRCA.sampleMap/Pathway_Paradigm_mRNA_And_Copy_Number
```

```
#>   TCGA.BRCA.sampleMap/Pathway_Paradigm_RNASeq
#>   TCGA.BRCA.sampleMap/Pathway_Paradigm_RNASeq_And_Copy_Number
#>   TCGA.BRCA.sampleMap/Pathway_Paradigm_mRNA
```

Sometimes we only know some keywords, `XenaScan()` can be used to scan all rows to detect if the keywords exist in `XenaData`.

```
x1 = XenaScan(pattern = "Blood")
x2 = XenaScan(pattern = "LUNG", ignore.case = FALSE)

x1 %>% XenaGenerate()
#> class: XenaHub
#> hosts():
#>   https://ucscpublic.xenahubs.net
#>   https://tcga.xenahubs.net
#> cohorts() (6 total):
#>   Acute lymphoblastic leukemia (Mullighan 2008)
#>   Connectivity Map
#>   Pediatric tumor (Khan)
#>   TARGET Acute Lymphoblastic Leukemia
#>   TCGA Acute Myeloid Leukemia (LAML)
#>   TCGA Pan-Cancer (PANCAN)
#> datasets() (34 total):
#>   mullighan2008_public/mullighan2008_500K_genomicMatrix
#>   mullighan2008_public/mullighan2008_SNP6_genomicMatrix
#>   cmap/rankMatrix_reverse
#>   ...
#>   TCGA.PANCAN.sampleMap/SNP6_genomicSegment
#>   TCGA.PANCAN.sampleMap/HiSeqV2_exon
x2 %>% XenaGenerate()
#> class: XenaHub
#> hosts():
#>   https://tcga.xenahubs.net
#> cohorts() (1 total):
#>   TCGA Lung Cancer (LUNG)
#> datasets() (13 total):
#>   TCGA.LUNG.sampleMap/HumanMethylation27
#>   TCGA.LUNG.sampleMap/HumanMethylation450
#>   TCGA.LUNG.sampleMap/Gistic2_CopyNumber_Gistic2_all_data_by_genes
#>   ...
#>   TCGA.LUNG.sampleMap/HiSeqV2_exon
#>   TCGA.LUNG.sampleMap/AgilentG4502A_07_3
```

Filter

There are too many datasets in `xe`, you can filter them by `XenaFilter` function. Regular expression can be used here.

```
(xe2 <- XenaFilter(xe, filterDatasets = "clinical"))
#> class: XenaHub
#> hosts():
#>   https://tcga.xenahubs.net
#> cohorts() (37 total):
#>   TCGA Acute Myeloid Leukemia (LAML)
#>   TCGA Adrenocortical Cancer (ACC)
#>   TCGA Bile Duct Cancer (CHOL)
#>   ...
#>   TCGA Thyroid Cancer (THCA)
#>   TCGA Uterine Carcinosarcoma (UCS)
#> datasets() (37 total):
#>   TCGA.LAML.sampleMap/LAML_clinicalMatrix
#>   TCGA.ACC.sampleMap/ACC_clinicalMatrix
#>   TCGA.CHOL.sampleMap/CHOL_clinicalMatrix
#>   ...
#>   TCGA.THCA.sampleMap/THCA_clinicalMatrix
#>   TCGA.UCS.sampleMap/UCS_clinicalMatrix
```

Then select `LUAD`, `LUSC` and `LUNG` 3 datasets.

```
xe2 <- XenaFilter(xe2, filterDatasets = "LUAD|LUSC|LUNG")
```

Pipe can be used here.

```
xe %>% XenaFilter(filterDatasets = "clinical") %>%
    XenaFilter(filterDatasets = "luad|lusc|lung")
#> class: XenaHub
#> hosts():
#>   https://tcga.xenahubs.net
#> cohorts() (3 total):
#>   TCGA Lung Adenocarcinoma (LUAD)
#>   TCGA Lung Cancer (LUNG)
#>   TCGA Lung Squamous Cell Carcinoma (LUSC)
#> datasets() (3 total):
#>   TCGA.LUAD.sampleMap/LUAD_clinicalMatrix
#>   TCGA.LUNG.sampleMap/LUNG_clinicalMatrix
#>   TCGA.LUSC.sampleMap/LUSC_clinicalMatrix
```

Browse datasets

Sometimes, you may want to check data before you query and download data.
A new feature `XenaBrowse` is implemented in **UCSCXenaTools**.

Create two XenaHub objects:

- `to_browse` - a XenaHub object contains a cohort and a dataset.
- `to_browse2` - a XenaHub object contains 2 cohorts and 2 datasets.

```
to_browse <- XenaGenerate(subset = XenaHostNames ==
    "tcgaHub") %>% XenaFilter(filterDatasets = "clinical") %>%
    XenaFilter(filterDatasets = "LUAD")

to_browse
#> class: XenaHub
#> hosts():
#>   https://tcga.xenahubs.net
#> cohorts() (1 total):
#>   TCGA Lung Adenocarcinoma (LUAD)
#> datasets() (1 total):
#>   TCGA.LUAD.sampleMap/LUAD_clinicalMatrix

to_browse2 <- XenaGenerate(subset = XenaHostNames ==
    "tcgaHub") %>% XenaFilter(filterDatasets = "clinical") %>%
    XenaFilter(filterDatasets = "LUAD|LUSC")

to_browse2
#> class: XenaHub
#> hosts():
#>   https://tcga.xenahubs.net
#> cohorts() (2 total):
#>   TCGA Lung Adenocarcinoma (LUAD)
#>   TCGA Lung Squamous Cell Carcinoma (LUSC)
#> datasets() (2 total):
#>   TCGA.LUAD.sampleMap/LUAD_clinicalMatrix
#>   TCGA.LUSC.sampleMap/LUSC_clinicalMatrix
```

`XenaBrowse()` function can be used to browse dataset/cohort links using your default web browser. At default, this function limit one dataset/cohort for preventing user to open too many links at once.

```
# This will open you web browser
XenaBrowse(to_browse)

XenaBrowse(to_browse, type = "cohort")
```

```
# This will throw error
XenaBrowse(to_browse2)
#> Error in XenaBrowse(to_browse2): This function limite 1 dataset to browse.
#>  Set multiple to TRUE if you want to browse multiple links.

XenaBrowse(to_browse2, type = "cohort")
#> Error in XenaBrowse(to_browse2, type = "cohort"): This function limite 1 cohort to browse.
#>  Set multiple to TRUE if you want to browse multiple links.
```

When you make sure you want to open multiple links, you can set `multiple` option to `TRUE`.

```
XenaBrowse(to_browse2, multiple = TRUE)
XenaBrowse(to_browse2, type = "cohort", multiple = TRUE)
```

Query

Create a query before downloading data.

```
xe2_query = XenaQuery(xe2)
#> This will check url status, please be patient.
xe2_query
#>                       hosts
#> 1 https://tcga.xenahubs.net
#> 2 https://tcga.xenahubs.net
#> 3 https://tcga.xenahubs.net
#>                                  datasets
#> 1 TCGA.LUAD.sampleMap/LUAD_clinicalMatrix
#> 2 TCGA.LUNG.sampleMap/LUNG_clinicalMatrix
#> 3 TCGA.LUSC.sampleMap/LUSC_clinicalMatrix
#>                                                                                  url
#> 1 https://tcga.xenahubs.net/download/TCGA.LUAD.sampleMap/LUAD_clinicalMatrix.gz
#> 2 https://tcga.xenahubs.net/download/TCGA.LUNG.sampleMap/LUNG_clinicalMatrix.gz
#> 3 https://tcga.xenahubs.net/download/TCGA.LUSC.sampleMap/LUSC_clinicalMatrix.gz
```

Download

Default, data will be downloaded to system temp directory. You can specify the path.

If the data exists, command will not run to download them, but you can force it by `force` option.

```
destdir = file.path(tempdir(), "test")
xe2_download = XenaDownload(xe2_query, destdir = destdir,
    trans_slash = TRUE)
#> All downloaded files will under directory /var/folders/mx/rfkl27z90c96wbmm3_kjk8c80000gn/T//RtmpYoj5
#> Downloading TCGA.LUAD.sampleMap__LUAD_clinicalMatrix.gz
#> Downloading TCGA.LUNG.sampleMap__LUNG_clinicalMatrix.gz
#> Downloading TCGA.LUSC.sampleMap__LUSC_clinicalMatrix.gz
#> Note file names inherit from names in datasets column
#>   and '/' all changed to '__'.
```

Of note, at default, the downloaded files will keep same directory structure as Xena. You can set `trans_slash` to TRUE, it will transform `/` in dataset id to `__`, this will make all downloaded files are under same directory.

Prepare

There are 4 ways to prepare data to R.

```
# way1: directory
cli1 = XenaPrepare(destdir)
names(cli1)
#> [1] "TCGA.LUAD.sampleMap__LUAD_clinicalMatrix.gz"
#> [2] "TCGA.LUNG.sampleMap__LUNG_clinicalMatrix.gz"
#> [3] "TCGA.LUSC.sampleMap__LUSC_clinicalMatrix.gz"
```

```
# way2: local files
cli2 = XenaPrepare(file.path(destdir, "TCGA.LUAD.sampleMap__LUAD_clinicalMatrix.gz"))
class(cli2)
#> [1] "spec_tbl_df" "tbl_df"      "tbl"
#> [4] "data.frame"
```

```
# way3: urls
cli3 = XenaPrepare(xe2_download$url[1:2])
names(cli3)
## [1] "LUSC_clinicalMatrix.gz" "LUNG_clinicalMatrix.gz"
```

```
# way4: xenadownload object
cli4 = XenaPrepare(xe2_download)
names(cli4)
#> [1] "TCGA.LUAD.sampleMap__LUAD_clinicalMatrix.gz"
#> [2] "TCGA.LUNG.sampleMap__LUNG_clinicalMatrix.gz"
#> [3] "TCGA.LUSC.sampleMap__LUSC_clinicalMatrix.gz"
```

From v0.2.6, `XenaPrepare()` can enable chunk feature when file is too big and user only need subset of file.

Following code show how to subset some rows or columns of files, `sample` is the name of the first column, user can directly use it in logical expression, `x` can be a representation of data frame user wanna do subset operation. More custom operation can be set as a function and pass to `callback` option.

```
# select rows which sample (gene symbol here) in "HIF3A" or "RNF17"
testRNA = UCSCXenaTools::XenaPrepare("~/Download/HiSeqV2.gz", use_chunk = TRUE, subset_rows = sample %i
# only keep 1 to 3 columns
testRNA = UCSCXenaTools::XenaPrepare("~/Download/HiSeqV2.gz", use_chunk = TRUE, select_cols = colnames(
```

Download TCGA data with readable options

getTCGAdata

`getTCGAdata` provides a more readable way for downloading TCGA (hg19 version, different from `gdcHub`) datasets, user can specify multiple options to select data and corresponding file type to download. Default this function will return a list include `XenaHub` object and selected datasets information. Once you are sure the datasets are exactly what you want, `download` can be set to `TRUE` to download the data.

Check arguments of `getTCGAdata`:

```
args(getTCGAdata)
#> function (project = NULL, clinical = TRUE, download = FALSE,
#>     forceDownload = FALSE, destdir = tempdir(), mRNASeq = FALSE,
#>     mRNAArray = FALSE, mRNASeqType = "normalized", miRNASeq = FALSE,
#>     exonRNASeq = FALSE, RPPAArray = FALSE, ReplicateBaseNormalization = FALSE,
#>     Methylation = FALSE, MethylationType = c("27K", "450K"),
#>     GeneMutation = FALSE, SomaticMutation = FALSE, GisticCopyNumber = FALSE,
#>     Gistic2Threshold = TRUE, CopyNumberSegment = FALSE, RemoveGermlineCNV = TRUE,
#>     ...)
#> NULL

# or run ??getTCGAdata to read documentation
```

Select one or more projects, default will select only clinical datasets:

```
getTCGAdata(c("UVM", "LUAD"))
#> $Xena
#> class: XenaHub
#> hosts():
#>   https://tcga.xenahubs.net
#> cohorts() (2 total):
#>   TCGA Lung Adenocarcinoma (LUAD)
#>   TCGA Ocular melanomas (UVM)
```

```
#> datasets() (2 total):
#>   TCGA.LUAD.sampleMap/LUAD_clinicalMatrix
#>   TCGA.UVM.sampleMap/UVM_clinicalMatrix
#>
#> $DataInfo
#> # A tibble: 2 x 20
#>   XenaHosts XenaHostNames XenaCohorts
#>   <chr>     <chr>         <chr>
#> 1 https://~ tcgaHub       TCGA Lung ~
#> 2 https://~ tcgaHub       TCGA Ocula~
#> # ... with 17 more variables:
#> #   XenaDatasets <chr>, SampleCount <chr>,
#> #   DataSubtype <chr>, Label <chr>,
#> #   Type <chr>, AnatomicalOrigin <chr>,
#> #   SampleType <chr>, Tags <chr>,
#> #   ProbeMap <chr>, LongTitle <chr>,
#> #   Citation <chr>, Version <chr>,
#> #   Unit <chr>, Platform <chr>,
#> #   ProjectID <chr>, DataType <chr>,
#> #   FileType <chr>

tcga_data = getTCGAdata(c("UVM", "LUAD"))

# only return XenaHub object
tcga_data$Xena
#> class: XenaHub
#> hosts():
#>   https://tcga.xenahubs.net
#> cohorts() (2 total):
#>   TCGA Lung Adenocarcinoma (LUAD)
#>   TCGA Ocular melanomas (UVM)
#> datasets() (2 total):
#>   TCGA.LUAD.sampleMap/LUAD_clinicalMatrix
#>   TCGA.UVM.sampleMap/UVM_clinicalMatrix

# only return datasets information
tcga_data$DataInfo
#> # A tibble: 2 x 20
#>   XenaHosts XenaHostNames XenaCohorts
#>   <chr>     <chr>         <chr>
#> 1 https://~ tcgaHub       TCGA Lung ~
#> 2 https://~ tcgaHub       TCGA Ocula~
#> # ... with 17 more variables:
```

```
#> #   XenaDatasets <chr>, SampleCount <chr>,
#> #   DataSubtype <chr>, Label <chr>,
#> #   Type <chr>, AnatomicalOrigin <chr>,
#> #   SampleType <chr>, Tags <chr>,
#> #   ProbeMap <chr>, LongTitle <chr>,
#> #   Citation <chr>, Version <chr>,
#> #   Unit <chr>, Platform <chr>,
#> #   ProjectID <chr>, DataType <chr>,
#> #   FileType <chr>
```

Set `download=TRUE` to download data, default data will be downloaded to system temp directory (you can specify the path with `destdir` option):

```
# only download clinical data
getTCGAdata(c("UVM", "LUAD"), download = TRUE)
```

Support Data Type and Options:

- clinical information: `clinical`
- mRNA Sequencing: `mRNASeq`
- mRNA microarray: `mRNAArray`
- miRNA Sequencing: `miRNASeq`
- exon Sequencing: `exonRNASeq`
- RPPA array: `RPPAArray`
- DNA Methylation: `Methylation`
- Gene mutation: `GeneMutation`
- Somatic mutation: `SomaticMutation`
- Gistic2 Copy Number: `GisticCopyNumber`
- Copy Number Segment: `CopyNumberSegment`

other data type supported by Xena cannot download use this function. Please refer to `downloadTCGA` function or `XenaGenerate` function.

NOTE: Sequencing data are all based on Illumina Hiseq platform, other platform (Illumina GA) data supported by Xena cannot download using this function. This is for building consistent data download flow. Mutation use `broad automated` version (except `PANCAN` use `MC3 Public Version`). If you wan to download other datasets, please refer to `downloadTCGA` function or `XenaGenerate` function.

Download any TCGA data by datatypes and filetypes

`downloadTCGA` function can be used to download any TCGA data supported by Xena, but in a way different from `getTCGAdata` function.

```
# download RNASeq data (use UVM as an example)
downloadTCGA(project = "UVM", data_type = "Gene Expression RNASeq",
    file_type = "IlluminaHiSeq RNASeqV2")
```

See the arguments:

```
args(downloadTCGA)
#> function (project = NULL, data_type = NULL, file_type = NULL,
#>     destdir = tempdir(), force = FALSE, ...)
#> NULL
```

Except `destdir` option, you only need to select three arguments for downloading data. Even throught the number is far less than `getTCGAdata`, it is more complex than the latter.

Before you download data, you need spare some time to figure out what data type and file type available and what your datasets have.

`availTCGA` can return all information you need:

```
availTCGA()
#> Note not all projects have listed data types and file types, you can use showTCGA function to check
#> $ProjectID
#>  [1] "LAML"     "ACC"      "CHOL"
#>  [4] "BLCA"     "BRCA"     "CESC"
#>  [7] "COADREAD" "COAD"     "UCEC"
#> [10] "ESCA"     "FPPP"     "GBM"
#> [13] "HNSC"     "KICH"     "KIRC"
#> [16] "KIRP"     "DLBC"     "LIHC"
#> [19] "LGG"      "GBMLGG"   "LUAD"
#> [22] "LUNG"     "LUSC"     "SKCM"
#> [25] "MESO"     "UVM"      "OV"
#> [28] "PANCAN"   "PAAD"     "PCPG"
#> [31] "PRAD"     "READ"     "SARC"
#> [34] "STAD"     "TGCT"     "THYM"
#> [37] "THCA"     "UCS"
#>
#> $DataType
#>  [1] "DNA Methylation"
#>  [2] "Gene Level Copy Number"
#>  [3] "Somatic Mutation"
#>  [4] "Gene Expression RNASeq"
#>  [5] "miRNA Mature Strand Expression RNASeq"
#>  [6] "Gene Somatic Non-silent Mutation"
#>  [7] "Copy Number Segments"
#>  [8] "Exon Expression RNASeq"
```

```
#>  [9] "Phenotype"
#> [10] "PARADIGM Pathway Activity"
#> [11] "Protein Expression RPPA"
#> [12] "Transcription Factor Regulatory Impact"
#> [13] "Gene Expression Array"
#> [14] "Signatures"
#> [15] "iCluster"
#>
#> $FileType
#>  [1] "Methylation27K"
#>  [2] "Methylation450K"
#>  [3] "Gistic2"
#>  [4] "wustl hiseq automated"
#>  [5] "IlluminaGA RNASeq"
#>  [6] "IlluminaHiSeq RNASeqV2 in percentile rank"
#>  [7] "IlluminaHiSeq RNASeqV2 pancan normalized"
#>  [8] "IlluminaHiSeq RNASeqV2"
#>  [9] "After remove germline cnv"
#> [10] "PANCAN AWG analyzed"
#> [11] "Clinical Information"
#> [12] "wustl automated"
#> [13] "Gistic2 thresholded"
#> [14] "Before remove germline cnv"
#> [15] "Use only RNASeq"
#> [16] "Use RNASeq plus Copy Number"
#> [17] "bcm automated"
#> [18] "IlluminaHiSeq RNASeq"
#> [19] "bcm curated"
#> [20] "broad curated"
#> [21] "RPPA"
#> [22] "bsgsc automated"
#> [23] "broad automated"
#> [24] "bcgsc automated"
#> [25] "ucsc automated"
#> [26] "RABIT Use IlluminaHiSeq RNASeqV2"
#> [27] "RABIT Use IlluminaHiSeq RNASeq"
#> [28] "RPPA normalized by RBN"
#> [29] "RABIT Use Agilent 244K Microarray"
#> [30] "wustl curated"
#> [31] "Use Microarray plus Copy Number"
#> [32] "Use only Microarray"
#> [33] "Agilent 244K Microarray"
#> [34] "IlluminaGA RNASeqV2"
```

```
#> [35] "bcm SOLiD"
#> [36] "RABIT Use IlluminaGA RNASeqV2"
#> [37] "RABIT Use IlluminaGA RNASeq"
#> [38] "RABIT Use Affymetrix U133A Microarray"
#> [39] "Affymetrix U133A Microarray"
#> [40] "MethylMix"
#> [41] "bcm SOLiD curated"
#> [42] "Gene Expression Subtype"
#> [43] "Platform-corrected PANCAN12 dataset"
#> [44] "Batch effects normalized"
#> [45] "MC3 Public Version"
#> [46] "TCGA Sample Type and Primary Disease"
#> [47] "RPPA pancan normalized"
#> [48] "Tumor copy number"
#> [49] "Genome-wide DNA Damage Footprint HRD Score"
#> [50] "TCGA Molecular Subtype"
#> [51] "iCluster cluster assignments"
#> [52] "iCluster latent variables"
#> [53] "RNA based StemnessScore"
#> [54] "DNA methylation based StemnessScore"
#> [55] "Pancan Gene Programs"
#> [56] "Immune Model Based Subtype"
#> [57] "Immune Signature Scores"
```

Note not all datasets have these property, `showTCGA` can help you to check it. It will return all data in TCGA, you can use following code in RStudio and search your data.

```
View(showTCGA())
```

OR you can use shiny app provided by `UCSCXenaTools` to search.
Run shiny by:

```
UCSCXenaTools::XenaShiny()
```

SessionInfo

```
sessionInfo()
#> R version 3.6.0 RC (2019-04-21 r76409)
#> Platform: x86_64-apple-darwin15.6.0 (64-bit)
#> Running under: macOS High Sierra 10.13.6
#>
#> Matrix products: default
```

```
#> BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
#> LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
#>
#> locale:
#> [1] zh_CN.UTF-8/zh_CN.UTF-8/zh_CN.UTF-8/C/zh_CN.UTF-8/zh_CN.UTF-8
#>
#> attached base packages:
#> [1] stats     graphics  grDevices utils
#> [5] datasets  methods   base
#>
#> other attached packages:
#> [1] dplyr_0.8.3         UCSCXenaTools_1.2.4
#> [3] pacman_0.5.1
#>
#> loaded via a namespace (and not attached):
#>  [1] Rcpp_1.0.1
#>  [2] pillar_1.4.2
#>  [3] compiler_3.6.0
#>  [4] formatR_1.7
#>  [5] later_0.8.0
#>  [6] tools_3.6.0
#>  [7] zeallot_0.1.0
#>  [8] digest_0.6.20
#>  [9] evaluate_0.14
#> [10] tibble_2.1.3
#> [11] pkgconfig_2.0.2
#> [12] rlang_0.4.0
#> [13] cli_1.1.0
#> [14] shiny_1.3.2
#> [15] curl_3.3
#> [16] yaml_2.2.0
#> [17] xfun_0.8
#> [18] httr_1.4.0
#> [19] stringr_1.4.0
#> [20] knitr_1.23
#> [21] vctrs_0.2.0
#> [22] hms_0.5.0
#> [23] shinydashboard_0.7.1
#> [24] tidyselect_0.2.5
#> [25] glue_1.3.1
#> [26] R6_2.4.0
#> [27] fansi_0.4.0
#> [28] rmarkdown_1.14
```

```
#> [29] readr_1.3.1
#> [30] purrr_0.3.2
#> [31] magrittr_1.5
#> [32] backports_1.1.4
#> [33] promises_1.0.1
#> [34] htmltools_0.3.6
#> [35] assertthat_0.2.1
#> [36] tint_0.1.2
#> [37] mime_0.7
#> [38] xtable_1.8-4
#> [39] httpuv_1.5.1
#> [40] utf8_1.1.4
#> [41] stringi_1.4.3
#> [42] crayon_1.3.4
```

Bug Report

I have no time to test if all conditions are right and all datasets can normally be downloaded. So if you have any question or suggestion, please open an issue on Github at https://github.com/ShixiangWang/UCSCXenaTools/issues.

Acknowledgement

This package is based on XenaR, thanks Martin Morgan for his work.

LICENSE