Data Process in R

Class 4

Kai Yao

May, 2016

Outline

- Review of Last Class
- Homework 2
- Supplements
- Movie Data Case
- In Class Exam

Outline

- Review of Last Class
- Homework 2
- Supplements
- Movie Data Case
- In Class Exam

How to Define Function

Returns the (parallel) maxima and minima of the input values.

Usage

```
max(..., na.rm = FALSE)
min(..., na.rm = FALSE)

pmax(..., na.rm = FALSE)
pmin(..., na.rm = FALSE)

pmax.int(..., na.rm = FALSE)
pmin.int(..., na.rm = FALSE)
```

Arguments

`...`	numeric or character arguments (see Note).
`na.rm`	a logical indicating whether missing values should be removed.

Details

`max` and `min` return the maximum or minimum of *all* the values present in their arguments, as `integer` if all are `logical` or `integer`, as `double` if all are numeric, and character otherwise.

If `na.rm` is `FALSE` an `NA` value in any of the arguments will cause a value of `NA` to be returned, otherwise `NA` values are ignored.

The minimum and maximum of a numeric empty set are `+Inf` and `-Inf` (in this order!) which ensures *transitivity*, e.g., `min(x1, min(x2)) == min(x1, x2)`. For numeric `x` `max(x) == -Inf` and `min(x) == +Inf` whenever `length(x) == 0` (after removing missing values if requested). However, `pmax` and `pmin` return `NA` if all the parallel elements are `NA` even for `na.rm = TRUE`.

`pmax` and `pmin` take one or more vectors (or matrices) as arguments and return a single vector giving the 'parallel' maxima (or minima) of the vectors. The first element of the result is the maximum (minimum) of the first elements of all the arguments, the second element of the result is the maximum (minimum) of the second elements of all the arguments and so on. Shorter inputs (of non-zero length) are recycled if necessary. Attributes (see `attributes`: such as `names` or `dim`) are copied from the first argument (if applicable).

`pmax.int` and `pmin.int` are faster internal versions only used when all arguments are atomic vectors and there are no classes: they drop all attributes. (Note that all versions fail for raw and complex vectors since these have no ordering.)

`max` and `min` are generic functions: methods can be defined for them individually or via the `Summary` group generic. For this to work properly, the arguments `...` should be unnamed, and dispatch is on the first argument.

By definition the min/max of a numeric vector containing an `NaN` is `NaN`, except that the min/max of any vector containing an `NA` is `NA` even if it also contains an `NaN`. Note that `max(NA, Inf) == NA` even though the maximum would be `Inf` whatever the missing value actually is.

Character versions are sorted lexicographically, and this depends on the collating sequence of the locale in use: the help for 'Comparison' gives details. The max/min of an empty character vector is defined to be character `NA`. (One could argue that as `""` is the smallest character element, the maximum should be `""`, but there is no obvious candidate for the minimum.)

Value

For `min` or `max`, a length-one vector. For `pmin` or `pmax`, a vector of length the longest of the input vectors, or length zero if one of the inputs had zero length.

The type of the result will be that of the highest of the inputs in the hierarchy integer < double < character.

For `min` and `max` if there are only numeric inputs and all are empty (after possible removal of `NA`s), the result is double (`Inf` or `-Inf`).

How to Define Function

Name

Re of the input values.

Usage

```
max(..., na.rm = FALSE)
min(..., na.rm = FALSE)

pmax(..., na.rm = FALSE)
pmin(..., na.rm = FALSE)

pmax.int(..., na.rm = FALSE)
pmin.int(..., na.rm = FALSE)
```

Arguments

`...`	numeric or character arguments (see Note).
`na.rm`	a logical indicating whether missing values should be removed.

Details

`max` and `min` return the maximum or minimum of *all* the values present in their arguments, as `integer` if all are `logical` or `integer`, as `double` if all are numeric, and character otherwise.

If `na.rm` is `FALSE` an `NA` value in any of the arguments will cause a value of `NA` to be returned, otherwise `NA` values are ignored.

The minimum and maximum of a numeric empty set are `+Inf` and `-Inf` (in this order!) which ensures *transitivity*, e.g., `min(x1, min(x2)) == min(x1, x2)`. For numeric `x` `max(x) == -Inf` and `min(x) == +Inf` whenever `length(x) == 0` (after removing missing values if requested). However, `pmax` and `pmin` return `NA` if all the parallel elements are `NA` even for `na.rm = TRUE`.

`pmax` and `pmin` take one or more vectors (or matrices) as arguments and return a single vector giving the 'parallel' maxima (or minima) of the vectors. The first element of the result is the maximum (minimum) of the first elements of all the arguments, the second element of the result is the maximum (minimum) of the second elements of all the arguments and so on. Shorter inputs (of non-zero length) are recycled if necessary. Attributes (see `attributes`: such as `names` or `dim`) are copied from the first argument (if applicable).

`pmax.int` and `pmin.int` are faster internal versions only used when all arguments are atomic vectors and there are no classes: they drop all attributes. (Note that all versions fail for raw and complex vectors since these have no ordering.)

`max` and `min` are generic functions: methods can be defined for them individually or via the `Summary` group generic. For this to work properly, the arguments `...` should be unnamed, and dispatch is on the first argument.

By definition the min/max of a numeric vector containing an `NaN` is `NaN`, except that the min/max of any vector containing an `NA` is `NA` even if it also contains an `NaN`. Note that `max(NA, Inf) == NA` even though the maximum would be `Inf` whatever the missing value actually is.

Character versions are sorted lexicographically, and this depends on the collating sequence of the locale in use: the help for 'Comparison' gives details. The max/min of an empty character vector is defined to be character `NA`. (One could argue that as `""` is the smallest character element, the maximum should be `""`, but there is no obvious candidate for the minimum.)

Value

For `min` or `max`, a length-one vector. For `pmin` or `pmax`, a vector of length the longest of the input vectors, or length zero if one of the inputs had zero length.

The type of the result will be that of the highest of the inputs in the hierarchy integer < double < character.

For `min` and `max` if there are only numeric inputs and all are empty (after possible removal of `NA`s), the result is double (`Inf` or `-Inf`).

How to Define Function

Name

Re of the input values.

Usage

```
max(..., na.rm = FALSE)
min(..., na.rm = FALSE)

pmax(..., na.rm =
pmin(..., na.

pmax.int(...,
pmin.int(..., na.
```

Input

Arguments

`...`	numeric or character arguments (see Note).
`na.rm`	a logical indicating whether missing values should be removed.

Details

`max` and `min` return the maximum or minimum of *all* the values present in their arguments, as `integer` if all are `logical` or `integer`, as `double` if all are numeric, and character otherwise.

If `na.rm` is `FALSE` an `NA` value in any of the arguments will cause a value of `NA` to be returned, otherwise `NA` values are ignored.

The minimum and maximum of a numeric empty set are `+Inf` and `-Inf` (in this order!) which ensures *transitivity*, e.g., `min(x1, min(x2)) == min(x1, x2)`. For numeric `x` `max(x) == -Inf` and `min(x) == +Inf` whenever `length(x) == 0` (after removing missing values if requested). However, `pmax` and `pmin` return `NA` if all the parallel elements are `NA` even for `na.rm = TRUE`.

`pmax` and `pmin` take one or more vectors (or matrices) as arguments and return a single vector giving the 'parallel' maxima (or minima) of the vectors. The first element of the result is the maximum (minimum) of the first elements of all the arguments, the second element of the result is the maximum (minimum) of the second elements of all the arguments and so on. Shorter inputs (of non-zero length) are recycled if necessary. Attributes (see `attributes`: such as `names` or `dim`) are copied from the first argument (if applicable).

`pmax.int` and `pmin.int` are faster internal versions only used when all arguments are atomic vectors and there are no classes: they drop all attributes. (Note that all versions fail for raw and complex vectors since these have no ordering.)

`max` and `min` are generic functions: methods can be defined for them individually or via the `Summary` group generic. For this to work properly, the arguments `...` should be unnamed, and dispatch is on the first argument.

By definition the min/max of a numeric vector containing an `NaN` is `NaN`, except that the min/max of any vector containing an `NA` is `NA` even if it also contains an `NaN`. Note that `max(NA, Inf) == NA` even though the maximum would be `Inf` whatever the missing value actually is.

Character versions are sorted lexicographically, and this depends on the collating sequence of the locale in use: the help for 'Comparison' gives details. The max/min of an empty character vector is defined to be character `NA`. (One could argue that as `""` is the smallest character element, the maximum should be `""`, but there is no obvious candidate for the minimum.)

Value

For `min` or `max`, a length-one vector. For `pmin` or `pmax`, a vector of length the longest of the input vectors, or length zero if one of the inputs had zero length.

The type of the result will be that of the highest of the inputs in the hierarchy integer < double < character.

For `min` and `max` if there are only numeric inputs and all are empty (after possible removal of `NA`s), the result is double (`Inf` or `-Inf`).

How to Define Function

Name

Re of the input values.

Usage

```
max(..., na.rm = FALSE)
min(..., na.rm = FALSE)

pmax(..., na.rm =
pmin(..., na.

pmax.int(...,
pmin.int(..., na.
```

Input

Arguments

`...`	numeric or character arguments (see Note).
`na.rm`	a logical indicating whether missing values should be removed.

Details

`max` and `min` return the maximum or minimum of *all* the values present in their arguments, as `integer` if all are `logical` or `integer`, as `double` if all are numeric, and character otherwise.

If `na.rm` is `FALSE` an `NA` value in any of the arguments will cause a value of `NA` to be returned, otherwise `NA` values are ignored.

The minimum and maximum of a numeric empty set are `+Inf` and `-Inf` (in this order!) which ensures *transitivity*, e.g., `min(x1, min(x2)) == min(x1, x2)`. For numeric `x` `max(x) == -Inf` and `min(x) == +Inf` whenever `length(x) == 0` (after removing missing values if requested). However, `pmax` and `pmin` return `NA` if all the parallel elements are `NA` even for `na.rm = TRUE`.

`pmax` and `pmin` take one or more vectors (or matrices) as arguments and return a single vector giving the 'parallel' maxima (or minima) of the vectors. The first element of the result is the maximum (minimum) of the first elements of all the arguments, the second element of the result is the maximum (minimum) of the second elements of all the arguments and so on. Shorter inputs (of non-zero length) are recycled if necessary. Attributes (see `attributes`: such as `names` or `dim`) are copied from the first argument (if applicable).

`pmax.int` and `pmin.int` are faster internal versions only used when all arguments are atomic vectors and there are no classes: they drop all attributes. (Note that all versions fail for raw and complex vectors since these have no ordering.)

`max` and `min` are generic functions: methods can be defined for them individually or via the `Summary` group generic. For this to work properly, the arguments `...` should be unnamed, and dispatch is on the first argument.

By definition th ctor containing an `NaN` is `NaN`, except that the min/max of any vector containing an `NA` is `NA` even if it also contains an `NaN`. Note that `max(NA, Inf) == NA` even though the maximum would be `Inf` what

Output

Characte , and this depends on the collating sequence of the locale in use: the help for 'Comparison' gives details. The max/min of an empty character vector is defined to be character `NA`. (One could argue that as lement, the maximum should be `""`, but there is no obvious candidate for the minimum.)

Value

For `min` or `max`, a length-one vector. For `pmin` or `pmax`, a vector of length the longest of the input vectors, or length zero if one of the inputs had zero length.

The type of the result will be that of the highest of the inputs in the hierarchy integer < double < character.

For `min` and `max` if there are only numeric inputs and all are empty (after possible removal of `NA`s), the result is double (`Inf` or `-Inf`).

How to Define Function

```
Function_Name = function(arg1, arg2…)

{

    #body of the function

    return(result);

}
```

The red parts are defined by programer!

How to Use Own Function

```
Function_Name = function(arg1, arg2)

{

    #body of the function

    return(result);

}

re = Function_Name(var1,var2)
```

Notice: relationships between var1, var2, re and arg1, arg2, result

Add

- rbind
- cbind

Delete

- Dataframe[-rownumber,]
- resave to another dataframe

Revise

- Dataframe[i,j] =
- Dataframe$var =
- Dataframe$var[i] =

Search

- Dataframe[i,j]
- Dataframe[c(1,2…),c(1,2…)]
- Dataframe[condition,]

e.g. Dataframe[Dataframe$var>100,]

Data Analysis Steps

1. Brief Overview
2. Data Clean
3. Description Analysis
4. Visualization
5. Modeling

Data Visualization

- plot
- hist
- sort

Outline

- Review of Last Class
- Homework 2
- Supplements
- Movie Data Case
- In Class Exam

Outline

- Review of Last Class
- Homework 2
- Supplements
- Movie Data Case
- In Class Exam

Supplements

- Install Packages
- Save RData and Use Across Platforms
- Date and Time Formats
- Checking Raw Data Before Data process
- Cross Table

Install Packages

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

Contributed Packages

Available Packages

Currently, the CRAN package repository features 8437 available packages.

Table of available packages, sorted by date of publication

Table of available packages, sorted by name

Installation of Packages

Please type `help("INSTALL")` or `help("install.packages")` in R for information on how to install packages from this repository. The manual R Installation and Administration (also contained in the R base sources) explains the process in detail.

CRAN Task Views allow you to browse packages by topic and provide tools to automatically install all packages for special areas of interest. Currently, 33 views are available.

Package Check Results

All packages are tested regularly on machines running Debian GNU/Linux, Fedora, OS X, Solaris and Windows.

The results are summarized in the check summary (some timings are also available). Additional details for Windows checking and building can be found in the Windows check summary.

Writing Your Own Packages

The manual Writing R Extensions (also contained in the R base sources) explains how to write new packages and how to contribute them to CRAN.

Repository Policies

The manual CRAN Repository Policy [PDF] describes the policies in place for the CRAN package repository.

Related Directories

Archive
: Previous versions of the packages listed above, and other packages formerly available.

Orphaned
: Packages with no active maintainer, see the corresponding README.

bin/windows/contrib
: Windows binaries of contributed packages

bin/macosx/contrib
: OS X Mavericks binaries of contributed packages

Example

googleVis: R Interface to Google Charts

R interface to Google Charts API, allowing users to create interactive charts based on data frames. Charts are displayed locally via the R HTTP help server. A modern browser with Internet connection is required and for some charts a Flash player. The data remains local and is not uploaded to Google.

Version:	0.5.10
Depends:	R (≥ 3.0.2)
Imports:	methods, RJSONIO, utils
Suggests:	shiny (≥ 0.4.0), httpuv (≥ 1.2.0), knitr (≥ 1.5)
Published:	2015-08-26
Author:	Markus Gesmann [aut, cre], Diego de Castillo [aut], Joe Cheng [ctb]
Maintainer:	Markus Gesmann <markus.gesmann at googlemail.com>
BugReports:	https://github.com/mages/googleVis/issues
License:	GPL-2 | GPL-3 [expanded from: GPL (≥ 2)]
URL:	https://github.com/mages/googleVis#googlevis
NeedsCompilation:	no
Citation:	googleVis citation info
Materials:	README NEWS
In views:	SpatioTemporal, WebTechnologies
CRAN checks:	googleVis results

Downloads:

Reference manual:	googleVis.pdf
Vignettes:	Using Roles and Intervals via googleVis Using Trendlines with googleVis Markdown example with knitr and googleVis Demonstration of googleVis Using Google Charts with R
Package source:	googleVis_0.5.10.tar.gz
Windows binaries:	r-devel: googleVis_0.5.10.zip, r-release: googleVis_0.5.10.zip, r-oldrel: googleVis_0.5.10.zip
OS X Mavericks binaries:	r-release: googleVis_0.5.10.tgz, r-oldrel: googleVis_0.5.10.tgz
Old sources:	googleVis archive

Reverse dependencies:

Reverse imports:	bayesPop, gtrendsR, MetaLandSim, mplot, RLumShiny, wppExplorer
Reverse suggests:	bayesTFR, Kmisc, Lahman, NanoStringNorm, spacetime, tfplot, vegdata

Example

googleVis: R Interface to Google Charts

R interface to Google Charts API, allowing users to create interactive charts based on data frames. Charts are displayed locally via the R HTTP help server. A modern browser with Internet connection is required and for some charts a Flash player. The data remains local and is not uploaded to Google.

Version:	0.5.10
Depends:	R (≥ 3.0.2)
Imports:	methods, RJSONIO, utils
Suggests:	shiny (≥ 0.4.0), httpuv (≥ 1.2.0), knitr (≥ 1.5)
Published:	2015-08-26
Author:	Markus Gesmann [aut, cre], Diego de Castillo [aut], Joe Cheng [ctb]
Maintainer:	Markus Gesmann <markus.gesmann at googlemail.com>
BugReports:	https://github.com/mages/googleVis/issues
License:	GPL-2 | GPL-3 [expanded from: GPL (≥ 2)]
URL:	https://github.com/mages/googleVis#googlevis
NeedsCompilation:	no
Citation:	googleVis citation info

Reference manual: googleVis.pdf

Vignettes:
- Using Roles and Intervals via googleVis
- Using Trendlines with googleVis
- Markdown example with knitr and googleVis
- Demonstration of googleVis
- Using Google Charts with R

Package source: googleVis_0.5.10.tar.gz

Save RData and Use Across Platforms

Save R Objects

Description

`save` writes an external representation of `R` objects to the specified file. The objects can be read back from the file at a later date by using the function `load` or `attach` (or `data` in some cases).

`save.image()` is just a short-cut for 'save my current workspace', i.e., `save(list = ls(all.names = TRUE), file = ".RData", envir = .GlobalEnv)`. It is also what happens with `q("yes")`.

Usage

```
save(..., list = character(),
     file = stop("'file' must be specified"),
     ascii = FALSE, version = NULL, envir = parent.frame(),
     compress = isTRUE(!ascii), compression_level,
     eval.promises = TRUE, precheck = TRUE)

save.image(file = ".RData", version = NULL, ascii = FALSE,
           compress = !ascii, safe = TRUE)
```

Date and Time Formats

Date Conversion Functions to and from Character

Description

Functions to convert between character representations and objects of class `"Date"` representing calendar dates.

Usage

```
as.Date(x, ...)
## S3 method for class 'character'
as.Date(x, format, ...)
## S3 method for class 'numeric'
as.Date(x, origin, ...)
## S3 method for class 'POSIXct'
as.Date(x, tz = "UTC", ...)

## S3 method for class 'Date'
format(x, ...)

## S3 method for class 'Date'
as.character(x, ...)
```

Arguments

`x`	An object to be converted.
`format`	A character string. If not specified, it will try `"%Y-%m-%d"` then `"%Y/%m/%d"` on the first non-`NA` element, and give an error if neither works. Otherwise, the processing is via `strptime`
`origin`	a Date object, or something which can be coerced by `as.Date(origin, ...)` to such an object.
`tz`	a time zone name.
`...`	Further arguments to be passed from or to other methods, including `format` for `as.character` and `as.Date` methods.

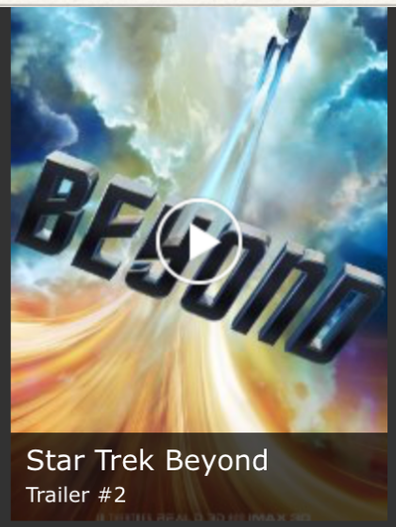
Checking Raw Data Before Data process

id	SubmitTime	TotalTime	Score	Name	Major
2	2016/5/8 17:19:20	280秒	60	张琬玥	大数据营销15
3	2016/5/8 17:19:28	282秒	60	鲁菁	大数据营销15
4	2016/5/8 17:19:40	305秒	60	周子轲	大数据营销15
5	2016/5/8 17:19:48	312秒	50	郑宇元	大数据营销15
6	2016/5/8 17:19:51	312秒	55	陈明准	市场营销14
7	2016/5/8 17:20:01	324秒	60	胡翔宇	大数据营销15
8	2016/5/8 17:20:11	331秒	55	王晨光	市场营销14
9	2016/5/8 17:20:26	351秒	60	吕训宇	大数据营销15
10	2016/5/8 17:20:36	355秒	50	黄菲	双培大数据营销15
11	2016/5/8 17:20:39	353秒	45	申羽峰	市场营销15
12	2016/5/8 17:20:50	371秒	60	蒋行健	大数据营销15

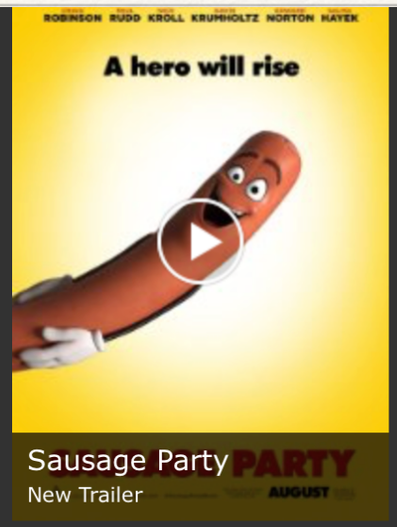
Outline

- Review of Last Class
- Homework 2
- Supplements
- Movie Data Case
- In Class Exam

IMDB



Star Trek Beyond
Trailer #2



Sausage Party
New Trailer

Ghostbusters
Trailer #2

Browse more trailers »

How Well Do You Know Your 'X-Men' Trivia?

Who almost played Nightcrawler? How many times has Hugh Jackman appeared as Wolverine? Learn 7 fun facts about the *X-Men* franchise that you need to know.

Learn some fun *X-Men* facts »

+ Angry Birds
+ Neighbors 2: Sorority Rising
+ The Nice Guys
+ Maggie's Plan — Limited
+ Weiner — Limited
+ Ma ma — Limited
+ Welcome to Happiness — Limited

See more opening this week »

Now Playing (Box Office)

+ Captain America: Civil War
  Weekend: $72.6M
+ The Jungle Book
  Weekend: $17.1M
+ Money Monster
  Weekend: $14.8M
+ The Darkness
  Weekend: $5.0M
+ Mother's Day
  Weekend: $3.3M

See more box office results »

Coming Soon

+ X-Men: Apocalypse
+ Alice Through the Looking Glass

Most Popular Feature Films Released In 2008

1-50 of 5,812 titles. Next »

Sort by: **Popularity▲** | A-Z | User Rating | Num Votes | US Box Office | Runtime | Year | US Release Date

1. 

Iron Man (2008) Add to Watchlist

7.9/10

After being held captive in an Afghan cave, a billionaire engineer creates a unique weaponized suit of armor to fight evil.

Dir: Jon Favreau With: Robert Downey Jr., Gwyneth Paltrow, Terrence Howard

Action | Adventure | Sci-Fi 126 mins. PG-13

2. 

The Dark Knight (2008) Add to Watchlist

9.0/10

When the menace known as the Joker wreaks havoc and chaos on the people of Gotham, the caped crusader must come to terms with one of the greatest psychological tests of his ability to fight injustice.

Dir: Christopher Nolan With: Christian Bale, Heath Ledger, Aaron Eckhart

Action | Crime | Thriller 152 mins. PG-13

3. 

Cloverfield (2008) Add to Watchlist

7.1/10

A group of friends venture deep into the streets of New York on a rescue mission during a rampaging monster attack.

Dir: Matt Reeves With: Mike Vogel, Jessica Lucas, Lizzy Caplan

Action | Adventure | Horror | Sci-Fi 85 mins. PG-13

Box Mojo

Box Office Mojo

Search Site

Search...

Social

Facebook

Twitter

Features

News

Release Sched.

Showtimes at IMDb

Box Office

Daily

TOP STORIES

'Angry Birds' Tops Friday With Estimated $11 Million, Headed Toward $40 Million Opening

by Brad Brevet

May 19 - While **Captain America: Civil War** crosses $1 billion worldwide it will attempt to hold off Sony's **Angry Birds** at this weekend's domestic box office. Meanwhile, it doesn't look like **Neighbors 2: Sorority Rising** will top its predecessor and Warner Bros. is hoping for the best with **The Nice Guys**, which has enjoyed the best reviews among the weekend's new wide releases.

Disney Makes History, Topping $1 Billion in Domestic Ticket Sales in Just 128 Days

FRIDAY ESTIMATES

1.	The Angry Birds Movie	$11,000,000
2.	Neighbors 2: Sorority Rising	$8,740,000
3.	Captain America: Civil War	$8,707,000

> VIEW FULL DAILY CHART

WORLDWIDE 2016

1.	Captain America: Civil War	$999,683,153
2.	Zootopia	$972,075,528
3.	Batman v Superman	$870,217,019
4.	The Jungle Book	$836,114,565
5.	Deadpool	$763,005,444

> VIEW FULL CHART

LATEST UPDATES

Example

Captain America: Civil War

Domestic Total as of May. 20, 2016: **$322,983,153 (Estimate)**

Distributor: **Buena Vista**	Release Date: **May 6, 2016**
Genre: **Action / Adventure**	Runtime: **2 hrs. 27 min.**
MPAA Rating: **PG-13**	Production Budget: **$250 million**

Summary | Daily | Weekend | **Weekly** | Foreign | Similar Movies

2016

Date (click to view chart)	Rank	Weekly Gross	% Change	Theaters / Change		Avg.	Gross-to-Date	Week #
May 6–12	1	$223,329,078	-	4,226	-	$52,846	$223,329,078	1
May 13–19	1	$90,947,075	-59.3%	4,226	-	$21,521	$314,276,153	2

Data Process

- Who is your clients or audience?
- What's your goals?
- What are the potential problems?
- What's the Dependent Variable (DV)
- Data clean
- Analysis
- Present the insights

Good Presentation

- Beautiful and Concise
- Good insights
- Reasonable story
- Attractive

Potential Interesting Findings

Outline

- Review of Last Class
- Homework 2
- Supplements
- Movie Data Case
- In Class Exam