

STOR 320.1

Data Import

Finding Data

- Built-in Datasets in R Packages
 - Example: NYC Flights
 - `>library(nycflights13)`
 - 5 Different Data Sets
 - More Comprehensive List
 - Vincent Arel-Bundock
 - [Link](#)
 - Packages
 - Data Name
 - Variable Information
 - CSV Links for Download
 - DOC Links for Details


Finding Data


- Online Websites
 - [Data.World](#)
 - Requires Sign-up
 - Search for Topic

The screenshot shows the Data.World search interface. At the top is a search bar with the text 'Education'. Below the search bar, on the left, is a sidebar with 'TYPE' and 'TAGS' sections. The 'TYPE' section has three options: 'Projects and datasets (2757)', 'Insights (10)', and 'People and organizations (27)'. The 'TAGS' section lists various tags with their counts: education (1550), schools (223), census (216), statistic (192), cso (191), statbank (191), lifelong learning (171), school (148), hunting (124), and doe (112). The main content area shows the top 10 results for 'Education'. The first result is 'Education @education' with a 'Follow' button. The next three results are projects: 'jzhao23/education' (updated Aug 8), 'alex-alex/Education' (updated May 10), and 'riccamini/Education' (updated Jul 30). Each project result includes a bookmark icon, a comment icon, and an 'OPEN' button.


Q Education

1-10 of 2,794


 **Education**
@education [Follow](#)

 **jzhao23/education**
Project • Updated Aug 8 [OPEN](#)

[Bookmark](#) [Comment](#)

 **alex-alex/Education**
Project • Updated May 10 [OPEN](#)

[Bookmark](#) [Comment](#)

 **riccamini/Education**
Project • Updated Jul 30 [OPEN](#)

[Bookmark](#) [Comment](#)

TYPE

- ☐ Projects and datasets (2757)
- ☐ Insights (10)
- ☐ People and organizations (27)

TAGS

- ☐ education (1550)
- ☐ schools (223)
- ☐ census (216)
- ☐ statistic (192)
- ☐ cso (191)
- ☐ statbank (191)
- ☐ lifelong learning (171)
- ☐ school (148)
- ☐ hunting (124)
- ☐ doe (112)

Finding Data

- Online Websites
 - [Data.World](#)
 - Inspect Data

▼ education/World University Rankings7 files, 6 tables

📁 world-university-rankings.zip

📄 world-university-rankings/cwu...

📄 world-university-rankings/edu...

📄 world-university-rankings/edu...

📄 world-university-rankings/scho...

📄 world-university-rankings/sha...

📄 world-university-rankings/time...

DATA SOURCES

Hide

▼ education/World University Rankings7 files, 6 tables

📁 world-university-rankings.zip

📄 world-university-rankings/cwu...

📄 world-university-rankings/edu...

📄 world-university-rankings/edu...

📄 world-university-rankings/scho...

📄 world-university-rankings/sha...

📄 world-university-rankings/time...

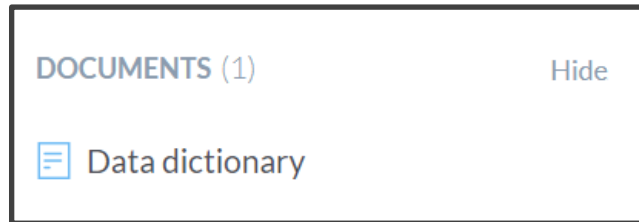
🏠 world-university-rankin... x




📄 world-university-rankings/cwurData.csv

	#	world_rank	📄 institution	📄 country	#	nationa
1		1	Harvard University	USA		
2		2	Massachusetts Institute of Technology	USA		
3		3	Stanford University	USA		
4		4	University of Cambridge	United Kingdom		
5		5	California Institute of Technology	USA		
6		6	Princeton University	USA		
7		7	University of Oxford	United Kingdom		

Finding Data

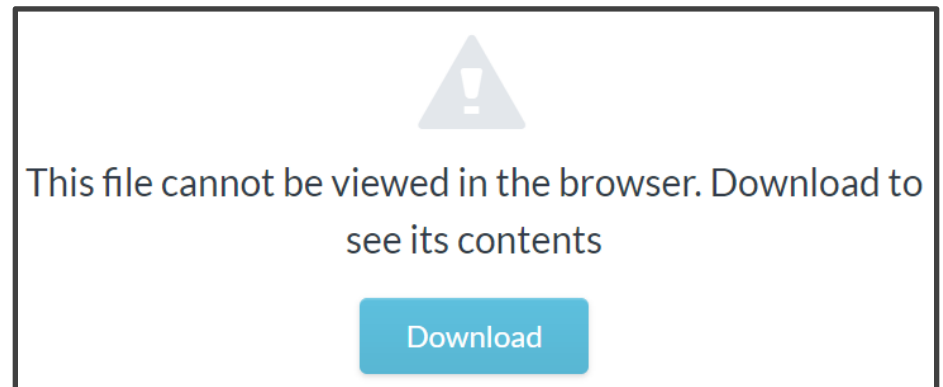
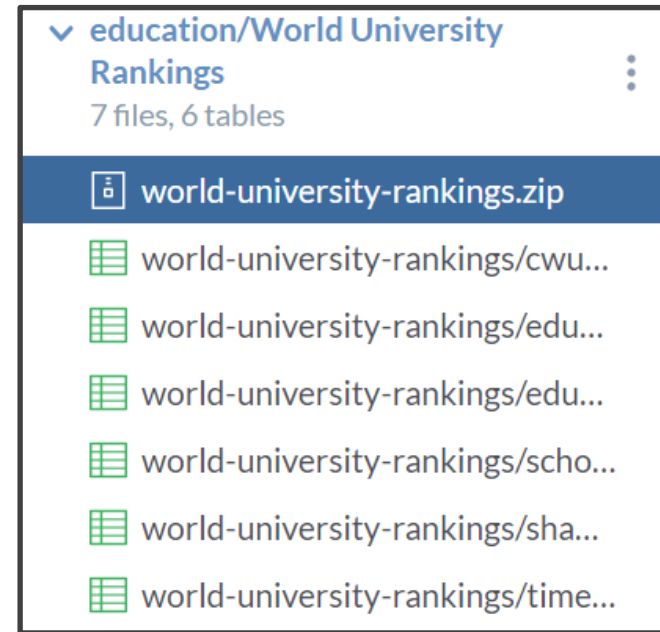
- Online Websites
 - [Data.World](#)
 - Read Data Dictionary



world-university-rankings/cwurData.csv	
Request more info	
#	world_rank
	institution
	country
#	national_rank
#	quality_of_education
#	alumni_employment
#	quality_of_faculty
#	publications
#	influence
#	citations
#	broad_impact
#	patents
#	score
	year

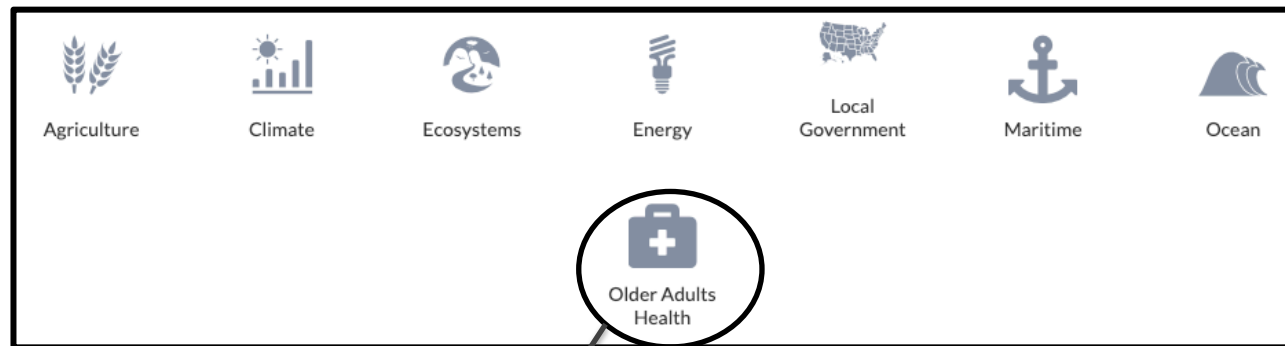
Finding Data

- Online Websites
 - [Data.World](https://data.world)
 - Download .zip Folder



Finding Data

- Online Websites
 - [Data.Gov](https://data.gov)
 - Logo
 - Topics List



Provisional Death Counts for Coronavirus Disease (COVID-19)


The provisional counts for coronavirus disease (COVID-19) deaths are based on a current flow of mortality data in the National Vital Statistics System. National provisional...



Finding Data


- Online Websites
 - [Data.Gov](https://data.gov)
 - Check Description


Housing Affordability Data System (HADS)

 Metadata Updated: March 8, 2017








The Housing Affordability Data System (HADS) is a set of files derived from the 1985 and later national American Housing Survey (AHS) and the 2002 and later Metro AHS. This system categorizes housing units by affordability and households by income, with respect to the Adjusted Median Income, Fair Market Rent (FMR), and poverty income. It also includes housing cost burden for owner and renter households. These files have been the basis for the worst case needs tables since 2001. The data files are available for public use, since they were derived from AHS public use files and the published income limits and FMRs. These dataset give the community of housing analysts the opportunity to use a consistent set of affordability measures.


Access & Use Information

 **Public:** This dataset is intended for public access and use.

 **License:** No license information was provided. If this work was prepared by an officer or employee of the United States government as part of that person's official duties it is considered a [U.S. Government Work](#).

Downloads & Resources

**Comma Separated Values File**
hads.html
 Link is ok      Openness score

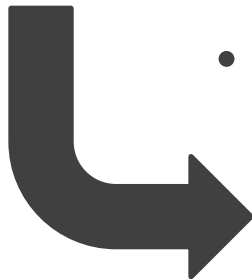
 **Download**

Finding Data

- Online Websites
 - [Data.Gov](#)
 - Find Documentation

[Download the HADS documentation file \(*.pdf, 159 KB\)](#)

The Housing Affordability Data System (HADS) is a set of housing-unit level datasets that measures the affordability of housing *units* and the housing cost burdens of *households*, relative to area median incomes, poverty level incomes, and Fair Market Rents. The purpose of these datasets is to provide housing analysts with consistent measures of affordability and burdens over a long period. The datasets are based on the American Housing Survey (AHS) national files from 1985 through 2009 and the metropolitan files from 2002 through 2009. Users can link records in HADS files to AHS records, allowing access to all of the AHS variables.



- Important Info About Data
 - Purpose of Data
 - Survey Data
 - Two Sets of Files
 - Years Included

Finding Data

- Online Websites
 - [Data.Gov](https://data.gov)
 - Download Links

HADS Data derived from AHS National Data

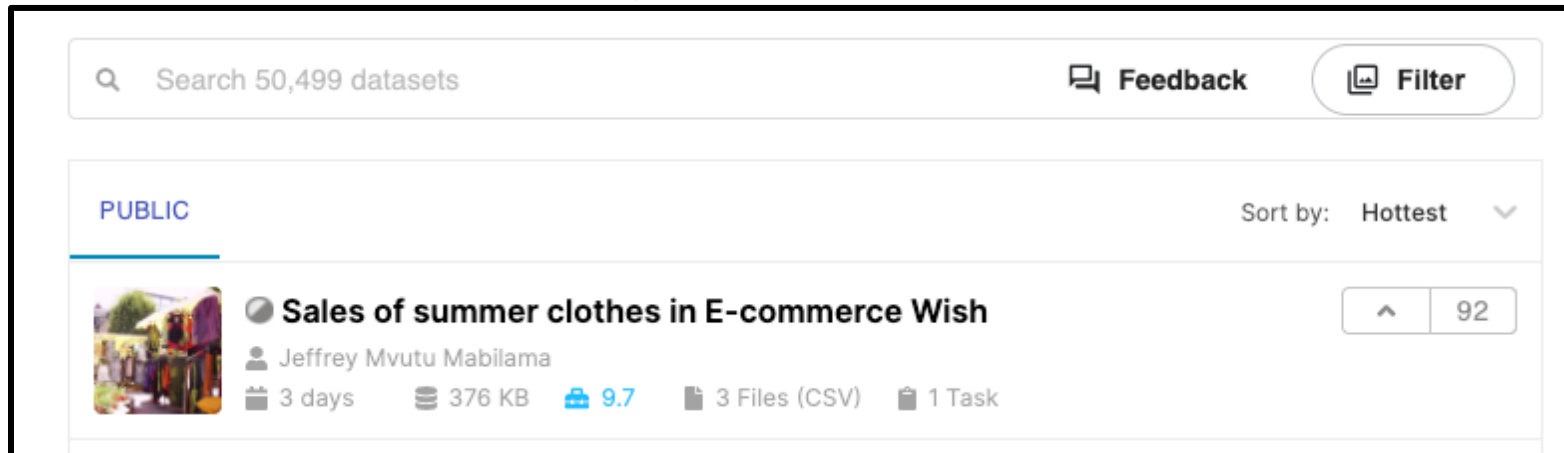
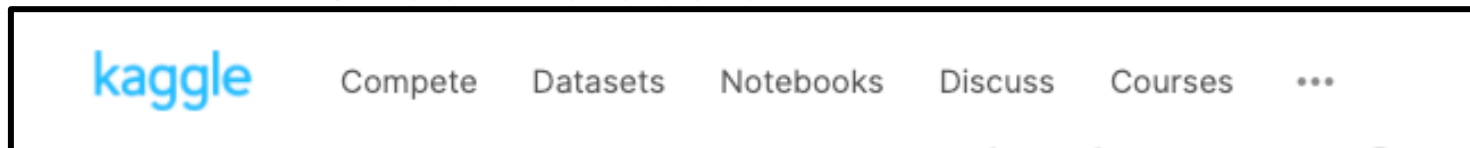
Year	ASCII version	SAS version
2013	*.zip (11.3 MB)	*.zip (18.8 MB)
2011	*.zip (22.3 MB)	*.zip (28.6 MB)

HADS Data derived from AHS Metro Data

Year	ASCII version	SAS version
2013	*.zip (9.4 MB)	*.zip (12.3 MB)
2009	Seattle Data (654 KB)	Seattle Data (727 KB)

Finding Data

- Online Websites
 - [Kaggle](#)
 - Requires Sign-up
 - Check Datasets



Finding Data

- Online Websites
 - [Kaggle](#)
 - Requires Sign-up
 - Overview and Question

Data [Overview](#) Kernels Discussion Activity

Competition Description



Ask a home buyer to describe their dream house, and they probably won't begin with the height of the basement ceiling or the proximity to an east-west railroad. But this playground competition's dataset proves that much more influences price negotiations than the number of bedrooms or a white-picket fence.

With 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, this competition challenges you to predict the final price of each home.

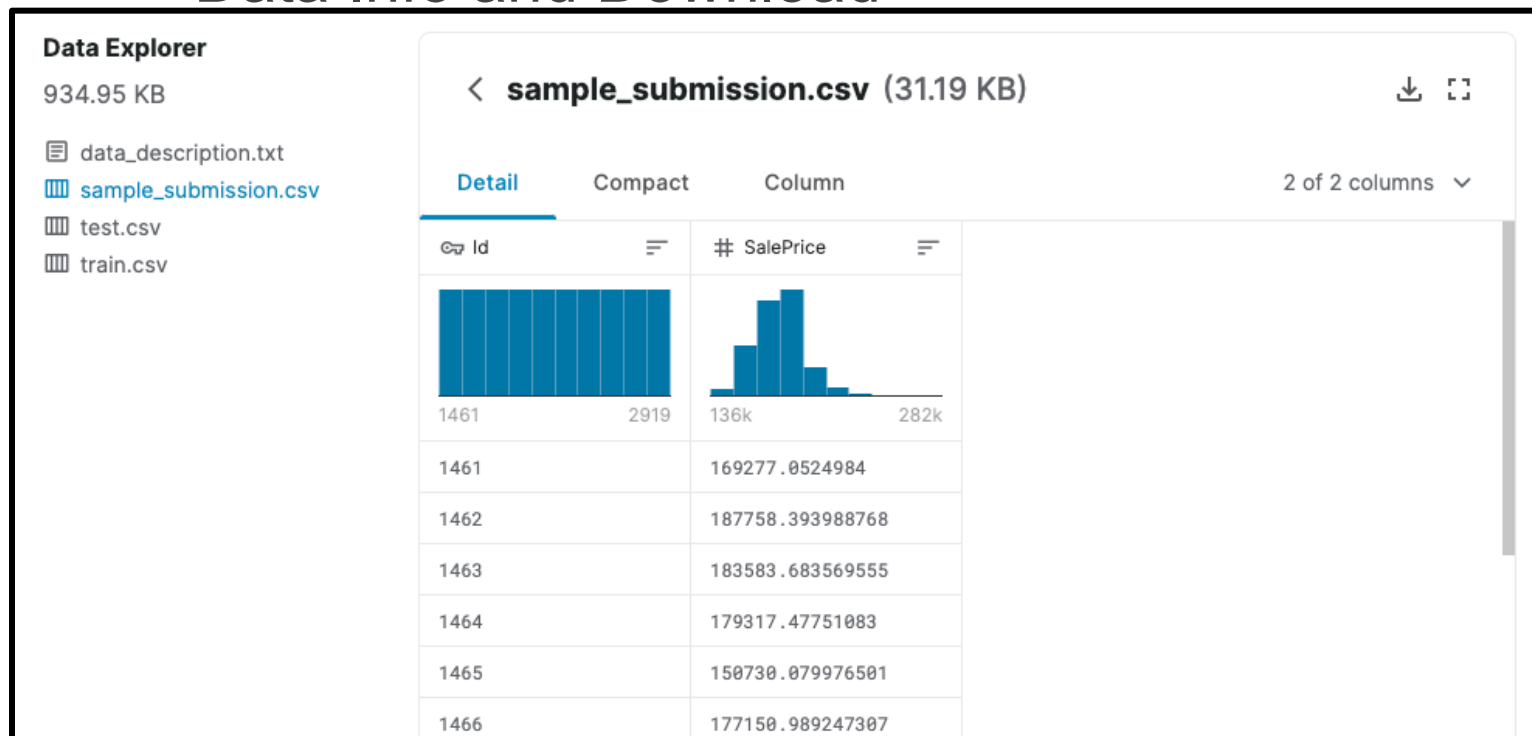
Goal

It is your job to predict the sales price for each house. For each Id in the test set, you must predict the value of the SalePrice variable.

Finding Data

- Online Websites
 - [Kaggle](#)
 - Requires Sign-up
 - Data Info and Download

[Data](#) Overview Kernels Discussion Activity



File Types

- Read Chapter 8
 - Package for Importing `>library(readr)`
 - Functions for Loading Data
- File Types
 - Different Delimiters
 - Comma, Tab, Space, Semicolon, Period
 - Different File Types
 - CSV – Comma
 - XLSX or XLS – Tab
 - TXT – Anything Possible
 - HTML – Anything Possible
 - Inspect Raw Data File

Data Import

- Importing CSV – Most Common
 - `read_csv()`

```
```{r}
UniRank=read_csv(file="/Users/yaoli/Documents/ACADEMIC/2020FALL/STOR320.1/L
ectures/Lecture06/Example/cwurData.csv",
 col_names=T)
glimpse(UniRank)
```
```

```
Observations: 2,198
Variables: 14
$ world_rank      <int> 1, 2, 3, 4, 5, 6, 7, 8, 9...
$ institution     <chr> "Harvard University", "Ma...
$ country        <chr> "USA", "USA", "USA", "Uni...
$ national_rank  <int> 1, 2, 3, 1, 4, 5, 2, 6, 7...
$ quality_of_education <int> 7, 9, 17, 10, 2, 8, 13, 1...
$ alumni_employment <int> 9, 17, 11, 24, 29, 14, 28...
$ quality_of_faculty <int> 1, 3, 5, 4, 7, 2, 9, 12, ...
$ publications    <int> 1, 12, 4, 16, 37, 53, 15,...
$ influence       <int> 1, 4, 2, 16, 22, 33, 13, ...
$ citations       <int> 1, 4, 2, 11, 22, 26, 19, ...
$ broad_impact    <int> NA, NA, NA, NA, NA, NA, N...
$ patents        <int> 5, 1, 15, 50, 18, 101, 26...
$ score          <dbl> 100.00, 91.67, 89.50, 86....
$ year           <int> 2012, 2012, 2012, 2012, 2...
```

- Auto Use of Column Names
- File Path Requires "/"
- Autodetects Variable Types

Data Import

- Importing CSV – column specification

```
```{r}
UniRank=read_csv(file="/Users/yaoli/Documents/ACADEMIC/2020FALL/STOR3
20.1/Lectures/Lecture06/Example/cwurData.csv",
 col_names=T)
glimpse(UniRank)
```
```

Parsed with column specification:

```
cols(
  world_rank = col_double(),
  institution = col_character(),
  country = col_character(),
  national_rank = col_double(),
  quality_of_education = col_double(),
  alumni_employment = col_double(),
  quality_of_faculty = col_double(),
  publications = col_double(),
  influence = col_double(),
  citations = col_double(),
  broad_impact = col_double(),
  patents = col_double(),
  score = col_double(),
  year = col_double()
)
```

Autodetect Info

- Always Check Tibble After Import
- Observe That Variable Types are What You Want

Data Import

- Other Types
 - `read_delim()` for General
 - XLS or XLSX

```
>library(readxl)
```

- Check Missing Values
 - See if NA's are Appropriately Recorded
 - Too Many NA's
 - Not Enough NA's
 - Crosscheck Raw Data and Data Documentation

Example

- HADS Data From Data.Gov

```
```{r,message=F}
Housing=read_csv(file="Example/thads2013n.txt")
head(Housing,5)
```
```

```
Housing=read_csv(file="Example/thads2013n.txt")
head(Housing,5)
```

```
## # A tibble: 5 x 99
##   CONTROL AGE1 METRO3 REGION LMED FMR L30 L50 L80 IPOV BEDRMS
##   <chr> <int> <chr> <chr> <int> <int> <int> <int> <int> <int> <int>
## 1 '10000~ 82 '3' '1' 73738 956 15738 26213 40322 11067 2
## 2 '10000~ 50 '5' '3' 55846 1100 17165 28604 45744 24218 4
## 3 '10000~ 53 '5' '3' 55846 1100 13750 22897 36614 15470 4
## 4 '10000~ 67 '5' '3' 55846 949 13750 22897 36614 13964 3
## 5 '10000~ 26 '1' '3' 60991 737 14801 24628 39421 15492 2
## # ... with 88 more variables: BUILT <int>, STATUS <chr>, TYPE <int>,
## # VALUE <int>, VACANCY <int>, TENURE <chr>, NUNITS <int>, ROOMS <int>,
## # WEIGHT <dbl>, PER <int>, ZINC2 <int>, ZADEQ <chr>, ZSMHC <int>,
## # STRUCTURETYPE <int>, OWNRENT <chr>, UTILITY <dbl>, OTHERCOST <dbl>,
## # COST06 <dbl>, COST12 <dbl>, COST08 <dbl>, COSTMED <dbl>, TOTSAL <int>
```

Example

- HADS Data From Data.Gov
 - Crosscheck

```

{r,message=F}
Housing2=read_csv(file="Example/thads2013n.txt") %>%
  select(BUILT,TOTSAL,VALUE,BURDEN)
Housing2
  
```

| BUILT
<dbl> | TOTSAL
<dbl> | VALUE
<dbl> | BURDEN
<dbl> |
|----------------|-----------------|----------------|-----------------|
| 2004 | 0 | 60000 | 0.22640619 |
| 2003 | 0 | 40000 | 0.07631935 |
| 1990 | 0 | 290000 | 0.04674121 |
| 1990 | 0 | 230000 | 0.12203823 |
| 2005 | 150000 | 260000 | 0.05674155 |
| 1990 | 0 | 280000 | 0.16979326 |
| 2004 | 69353 | 110000 | 0.17742005 |
| 1980 | -9 | -6 | -9.00000000 |
| 2007 | 200000 | 330000 | 0.12745657 |
| 2012 | 100000 | 90000 | 0.09565243 |

Errors or Missing
Should Become NA

Total salary income (**TotSal**) is useful for identifying the “working poor” and measuring the labor force attachment of a household. This variable is simply the sum of wage and salary income (Sal) over all members of the household.¹⁵

VALUE

Current market value of unit

BURDEN

Housing cost as a fraction of income

URL to R

- Benefit: Don't Need Data on PC
- Problem: Links Change
- Example

Music CSV Library

From the [CORGIS Dataset Project](#)

By Ryan Whitcomb (rwhit94@vt.edu)

Version 1, created 5-18-16

Tags: music, songs, artists, creativity, media



Overview

This library comes from the Million Song Dataset, which contains about one million popular contemporary songs. The dataset was created by LabROSA, a laboratory working towards intelligent music analysis at the National Science Foundation of America (NSF) to create a commercial size while promoting further research. The dataset contains standard information about the songs such as artist name, album, and more advanced information; for example, the length of the song, the fade in to the song was.


Downloads

Download all of the following files.

1. [music.csv](#) 

Downloads

Download all of the following files.

1. [music.csv](#) 

Field Description

| Key | | Comment |
|---------------------|--|---------|
| artist.hottnesss | | |
| artist.id | | |
| artist.name | | |
| artist_mbtags | | |
| artist_mbtags_count | | |

- Open in new tab
- Open in new window
- Open in new InPrivate window
- Save target as
- Copy link
- Add to reading list
- Search the web for "music.csv"
- Ask Cortana about "music.csv"
- View source
- Inspect element

URL to R

- Example

Field Descriptions

| Key | List of... | Comment | Example Value |
|---------------------|-------------|---------|----------------------|
| artist.hotttnesss | Real number | | 0.401997543 |
| artist.id | String | | "ARD7TVE1187B998FB1" |
| artist.name | String | | "Casual" |
| artist_mbtags | String | | " " |
| artist_mbtags_count | Real number | | 0.0 |

```
```{r,message=F}  
FreshBeats=read_csv(url("https://corgis-edu.github.io/corgis/datasets/csv/
music/music.csv"))
```

```
FreshBeats %>%
 filter(artist.name=="Black Eyed Peas") %>%
 arrange(desc(artist.hotttnesss)) %>%
 select(artist.hotttnesss,artist.name,artist.terms)
```
```

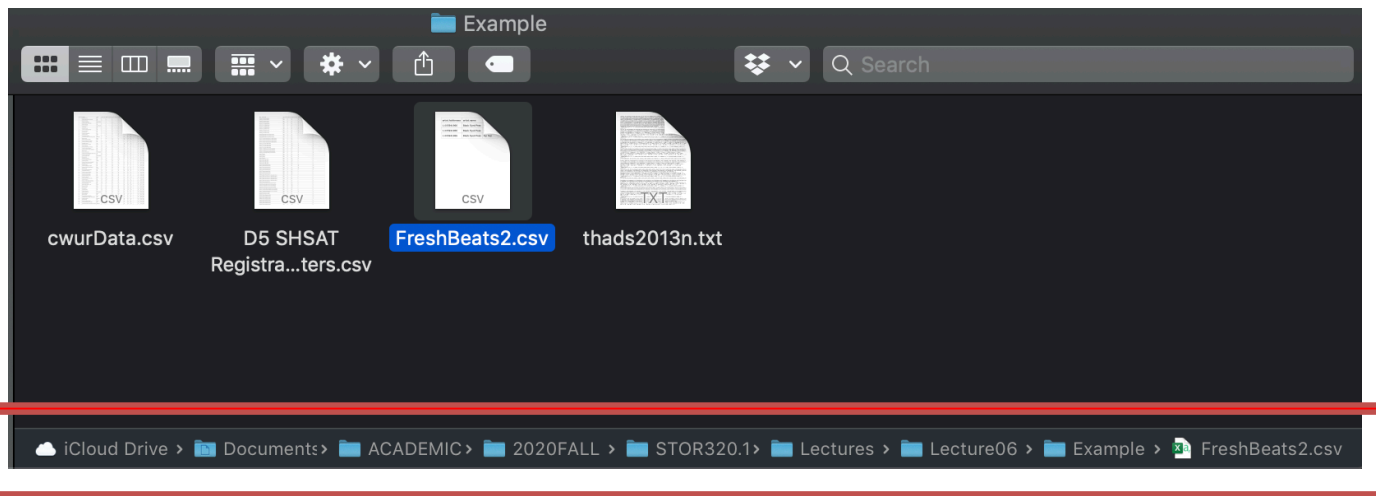
| artist.hotttnesss
<dbl> | artist.name
<chr> | artist.terms
<chr> |
|----------------------------|----------------------|-----------------------|
| 1.005942 | Black Eyed Peas | hip hop |
| 1.005942 | Black Eyed Peas | hip hop |
| 1.005942 | Black Eyed Peas | hip hop |

Writing Data

- `write_csv()`
 - Saves R Tibble to Computer

```
{r}  
setwd("Example")  
write_csv(FreshBeats, "FreshBeats.csv")
```

| | A | B | C | D | E | F |
|---|------------------|---------------------|--------------|---|---|---|
| 1 | artist.hotttness | artist.name | artist.terms | | | |
| 2 | 1.00594197 | Black Eyed Phip hop | | | | |
| 3 | 1.00594197 | Black Eyed Phip hop | | | | |
| 4 | 1.00594197 | Black Eyed Phip hop | | | | |
| 5 | | | | | | |



Tibble

- Read Chapter 7
 - Tribbles
 - Tibbles vs. data.frame

```
DATA=tribble(  
  ~x, ~y, ~z,  
  #---/---/---  
  "a", 2, 3.6,  
  "b", 1, 8.5  
)  
DATA
```

```
## # A tibble: 2 x 3  
##   x         y     z  
##   <chr> <dbl> <dbl>  
## 1 a         2   3.6  
## 2 b         1   8.5
```

- Subsetting Info

```
#Extract by Variable Name  
DATA$x
```

```
## [1] "a" "b"
```

```
DATA[[1]]
```

```
## [1] "a" "b"
```

```
DATA[["y"]]
```

```
## [1] 2 1
```

```
DATA[,c("x", "y")]
```

```
## # A tibble: 2 x 2  
##   x         y  
##   <chr> <dbl>  
## 1 a         2  
## 2 b         1
```

```
DATA[,3]
```

```
## # A tibble: 2 x 1  
##   `:`(  
##   <dbl>  
## 1   3.6  
## 2   8.5
```

```
DATA[2,]
```

```
## # A tibble: 1 x 3  
##   x         y `:`(  
##   <chr> <dbl> <dbl>  
## 1 b         1   8.5
```

```
DATA[2,2:3]
```

```
## # A tibble: 1 x 2  
##   y `:`(  
##   <dbl> <dbl>  
## 1     1   8.5
```