

# Package ‘qmj’

May 9, 2026

**Title** Quality Scores for the Russell 3000

**Version** 0.2.1

**Description** Produces quality scores for each of the US companies from the Russell 3000, following the approach described in “Quality Minus Junk” (Asness, Frazzini, & Pedersen, 2013) <<http://www.aqr.com/library/working-papers/quality-minus-junk>>. The package includes datasets for users who wish to view the most recently uploaded quality scores. It also provides tools to automatically gather relevant financials and stock price information, allowing users to update their data and customize their universe for further analysis.

**Imports** quantmod(>= 0.4-3), dplyr, reticulate, rlang

**Depends** R (>= 3.5.0)

**Suggests** testthat

**License** GPL-3

**BugReports** <https://github.com/anttsou/qmj/issues>

**LazyData** true

**LazyDataCompression** xz

**RoxygenNote** 7.3.2

**Encoding** UTF-8

**Config/reticulate** list(packages = list(list(package = “yfinance”)))

**NeedsCompilation** no

**Author** Anthony Tsou [aut],  
Eugene Choe [aut],  
David Kane [aut],  
Ryan Kwon [aut],  
Yanrong Song [aut, cre],  
Zijie Zhu [aut]

**Maintainer** Yanrong Song <yrsong129@gmail.com>

**Repository** CRAN

**Date/Publication** 2025-01-15 18:20:02 UTC

## Contents

clean_downloads	2
companies_r3k16	3
financials_r3k16	4
get_companies	5
get_info	6
get_prices	7
install_yfinance	8
market_data	9
market_growth	10
market_payouts	11
market_profitability	12
market_safety	13
prices_r3k16	14
qmj	15
quality_r3k16	16
tidyinfo	16
tidy_balancesheets	17
tidy_cashflows	18
tidy_helper	19
tidy_incomestatemnts	19
tidy_prices	20

<b>Index</b>	<b>22</b>
--------------	-----------

---

clean_downloads	<i>Removes downloaded temporary files.</i>
-----------------	--

---

### Description

clean\_downloads removes files that get\_info and get\_prices temporarily store when progress is interrupted while updating. Because the temporarily stored data may become irrelevant over time, clean\_downloads removes these files so that get\_info and get\_prices will download completely fresh sets of data for a given data frame of companies.

### Usage

```
clean_downloads(x = qmj::companies_r3k16)
```

### Arguments

x                    A data frame of companies. Must have a ticker column.

### Details

The clean\_downloads() function will also automatically remove any temporarily stored data for the S&P 500, with the stock ticker ^GSPC.

**Value**

A logical vector Where the (2i-1)th and (2i)th element corresponds to whether or not a temporary financial and/or price file, respectively, was found and removed for the ith company provided.

For example, if AAPL was our 3rd company, for which we had not partially downloaded financial data, but did have temporary price data, the 5th and 6th elements of the logical vector would be FALSE and TRUE, respectively.

The last two indices refer to the S&P 500 temporary data.

**See Also**

[get\\_prices](#)

[get\\_info](#)

**Examples**

```
clean_downloads()
```

---

companies\_r3k16

*A list of all companies in the Russell 3000 Index*

---

**Description**

Stores the names and tickers for all companies in the Russell 3000 Index as of January 2016. The list from which the data was culled was last updated 2015/06/26.

**Format**

A data frame with approximately 3000 rows and 2 variables.

- name = The name of the company. Of class "character".
- ticker = The ticker of the company. Of class "character".

**Details**

The Russell 3000 Index is an equity index that tracks the performance of the "3000" (this number may actually vary from year to year, but is always in the neighborhood of 3000) largest US companies as measured by market cap. The component companies that make up this index are reconstituted once a year, usually between May and June. At this reconstitution, all companies are reranked based on their market caps for the year, and any companies which become "ineligible" by, for example, going bankrupt, becoming acquired, or becoming private, are replaced at this time.

This Index was chosen due to the size of its component companies (which mitigates the likelihood of erroneous items, such as a tiny company doubling in profitability despite there being little absolute change), this package's reliance on US-centric data sources, and to produce items which are more likely to interest the user.

Companies\_r3k16 crucially provides tickers to many functions in the package, allowing the package to connect financial statements and price information to a specific company. It is also the basis of the many "get" functions of the package, which retrieves and then formats data from the web. The Companies\_r3k16 data set is the "base" data that produces financials, prices, and ultimately quality scores.

### Source

<https://www.lseg.com/en/ftse-russell>

### See Also

[financials\\_r3k16](#)

[prices\\_r3k16](#)

---

financials_r3k16	<i>Financial statements of all companies in the Russell 3000 index for the past four years</i>
------------------	--

---

### Description

A data frame containing all annual financial statements (balancesheets, cashflows, and income statements) for the past four years if available. For a description of the Russell 3000 index, as well as why it was used for this package, see [companies\\_r3k16](#). Last updated 2016/01/06.

### Format

A data frame with approximately 12000 rows and 23 variables

- AM = Amortization, of class "character".
- CWC = Changes in Working Capital, of class "character".
- CX = Capital Expenditures, of class "character".
- DIVC = Dividends per Share, of class "character".
- DO = Discontinued Operations, of class "character".
- DP.DPL = Depreciation/Depletion, of class "character".
- GPROF = Gross Profits, of class "character".
- IAT = Income After Taxes, of class "character".
- IBT = Income Before Taxes, of class "character".
- NI = Net Income, of class "character".
- NINT = Interest and Expense - Net Operating, of class "character".
- NRPS = Non-redeemable Preferred Stock, of class "character".
- RPS = Redeemable Preferred Stock, of class "character".
- TA = Total Assets, of class "character".

- TCA = Total Current Assets, of class "character".
- TCL = Total Current Liabilities, of class "character".
- TCSO = Total Common Shares Outstanding, of class "character".
- TD = Total Debt, of class "character".
- TL = Total Liabilities, of class "character".
- TLSE = Total Liabilities and Shareholders' Equity, of class "character".
- TREV = Total Revenue, of class "character".

### Details

Some companies may store "weird" data, such as having information solely for the years 1997-2001, or by having multiple annual reports within the same year (such as one report being filed in March of 2013, and another filed in December of 2013). In the case of companies reporting multiple annual data from the same year, the years of their reports are suffixed with their order. For example, GOOG may have data from 2013.1, 2013.2, 2012.3, 2011.4. This means Google's most recent data set is from 2013 (2013.1), another data set was published in 2013 (2013.2), and the remaining years are also suffixed for convenience.

The main purpose of `financials_r3k16` is to provide key information for each company in order to calculate each of the quality component scores (profitability, growth, safety, and payouts). For every ticker in the `companies_r3k16` data set, `financials_r3k16` will try to store the most recent four years of annual data, though this may vary based on availability.

### Source

Google & Yahoo Finance, accessed through `quantmod` & `yfinance`

### See Also

[companies\\_r3k16](#)

[prices\\_r3k16](#)

---

`get_companies`

*Builds a companies data frame from a text file.*

---

### Description

`get_companies` reads in the contents of a text file created from the pdf of company names and tickers given by the Russell 3000 Index.

### Usage

```
get_companies(filepath = system.file("extdata/companies.txt", package = "qmj"))
```

**Arguments**

filepath            Specifies the filepath of the text file containing the company names and tickers of interest. May be either absolute or relative to working directory.

**Details**

The user must copy and paste the contents of the Russell 3000 Index into a text file for this function to process the data correctly. Simply select all of the component list and paste the contents into an empty document with the .txt extension. The list may be found [here](#).

If you wish to use your own text file of companies for get\_companies to process, create a text file containing each company on a separate line. Every word and ticker must be capitalized, and the ticker must be the last word, separated by a space, on each line.

get\_companies by default uses a text file created from the Russell 3000 Index in the package.

**Value**

data.frame of companies info

**Functions**

- get\_companies(): function splits by space and grabs everything before the last word as the name and has the last word as the ticker. Which chunk it returns is determined by the is\_name variable.

**Examples**

```
get_companies()
```

---

get\_info

*Gets raw financial statements from Google Finance.*

---

**Description**

get\_info grabs annual financial data for a given data frame of companies.

**Usage**

```
get_info(companies = qmj::companies_r3k16)
```

**Arguments**

companies            A data frame of companies. Must have a ticker column.

## Details

For each ticker in the data frame of companies, `get_info` grabs financial data using the `quantmod` package and generates a list with three sub-lists. Also writes `.RData` files to the user's temporary directory. If cancelled partway through, `get_info` is able to find and re-read this data, quickly resuming its progress. Once complete, `get_info` deletes all used temporary data.

Parameter data frame defaults to provided `companies_r3k16` data set if not specified.

## Value

A list with three elements. Each element is a list containing all financial documents of a specific type for each company. These lists are, in order, all cash flow statements, all income statements, and all balance sheets.

## See Also

[get\\_prices](#)

[clean\\_downloads](#)

[tidyinfo](#)

## Examples

```
# Takes more than 10 secs
if (reticulate::py_module_available("yfinance"))
  get_info(companies_r3k16[companies_r3k16$ticker %in% c("AAPL", "AMZN"), ])
```

---

`get_prices`

*Grab daily prices and price returns for the previous two years.*

---

## Description

`get_prices` grabs price-related data for a given data frame of companies and returns a matrix-like object containing relevant price data.

## Usage

```
get_prices(companies = qmj::companies_r3k16)
```

## Arguments

`companies` A data frame of company names and tickers.

**Details**

`get_prices` is also able to write `.RData` files to the user's temporary directory. If canceled partway through, the function is able to find and re-read this data to resume progress. Once complete, the function deletes all used temporary data.

Parameter defaults to provided data set of companies if empty.

**Value**

A matrix-like object containing relevant price data. The rows of the matrix are dates in the international standard of YYYY-MM-DD. Each column specifies what data it covers in the form of TICKER.DATA, with the exception of price returns, which are stored as `pret.#`, where `#` refers to the `i`-th company given.

The first company calculated is always the S&P 500, and its price return column is simply `'pret'`.

**Functions**

- `get_prices()`: Calculates price returns for an xts object.

**See Also**

[get\\_info](#)

[clean\\_downloads](#)

[tidy\\_prices](#)

**Examples**

```
get_prices(companies_r3k16[companies_r3k16$ticker %in% c("AAPL", "AMZN"), ])
```

---

install\_yfinance

*Install yfinance and dependencies*

---

**Description**

`'install_yfinance()'` installs just the `yfinance` python package and its direct dependencies. Users may be asked to install `miniconda` for the python installations. Even if you first decline it, you can later install `miniconda` by running `['reticulate::install_miniconda()']`

**Usage**

```
install_yfinance(..., envname = "r-qmj", new_env = identical(envname, "r-qmj"))
```

**Arguments**

...	other arguments passed to [ <code>reticulate::py_install()</code> ]
envname	The name, or full path, of the environment in which Python packages are to be installed. When NULL (the default), the active environment as set by the <code>RETICULATE_PYTHON_ENV</code> variable will be used; if that is unset, then the <code>r-reticulate</code> environment will be used.
new_env	Delete 'envname' if it already exists

**Value**

No return value, called for installing Python yfinance module

---

market_data	<i>Produces component and quality scores.</i>
-------------	---

---

**Description**

Calculates market growth, payouts, safety, and profitability of our list of companies for later processing.

**Usage**

```
market_data(
  companies = qmj::companies_r3k16,
  financials = qmj::financials_r3k16,
  prices = qmj::prices_r3k16
)
```

**Arguments**

companies	A data frame of company names and tickers.
financials	A data frame containing financial information for the given companies.
prices	A data frame containing the daily market closing prices and returns.

**Details**

All parameters default to package data sets and must be formatted similarly to a data frame produced by [tidy\\_prices](#) and [tidyinfo](#).

**Value**

A data frame containing company names, tickers, profitability z-scores, growth z-scores, safety z-scores, payout z-scores, and quality z-scores. Organized by quality in descending order.

data.frame of all market data

**See Also**

[market\\_profitability](#)  
[market\\_growth](#)  
[market\\_safety](#)  
[market\\_payouts](#)

**Examples**

```
# Takes more than 10 secs  
market_data(companies_r3k16[companies_r3k16$ticker %in% c("AAPL"), ])
```

---

market_growth	<i>Collects growth z-scores for companies</i>
---------------	---

---

**Description**

Given a data frame of companies (names and tickers) and a data frame of financial statements, calculates GPOA, ROE, ROA, CFOA, GMAR, ACC over a four-year time span and determines the z-score of overall growth for each company based on the paper Quality Minus Junk (Asness et al.) in Appendix page A2.

**Usage**

```
market_growth(  
  companies = qmj::companies_r3k16,  
  financials = qmj::financials_r3k16  
)
```

**Arguments**

companies      A data frame of company names and tickers.  
financials      A data frame containing financial statements for every company.

**Value**

data.frame of market growth values

**See Also**

[market\\_data](#)  
[market\\_profitability](#)  
[market\\_safety](#)  
[market\\_payouts](#)

**Examples**

```
market_growth(companies_r3k16[1,], financials_r3k16)
```

---

market_payouts	<i>Collects payout z-scores for companies</i>
----------------	---

---

**Description**

Given a data frame of companies (names and tickers) and a data frame of financial statements, calculates EISS, DISS, NPOP and determines the z-score of overall payout for each company based on the paper Quality Minus Junk (Asness et al.) in Appendix page A3-4.

**Usage**

```
market_payouts(  
  companies = qmj::companies_r3k16,  
  financials = qmj::financials_r3k16  
)
```

**Arguments**

companies	A data frame of company names and tickers. Requires a 'ticker' column. Defaults to the provided companies data set.
financials	A data frame containing financial statements for every company. Defaults to the provided financials data set.

**Value**

data.frame of market payouts values

**Functions**

- `market_payouts()`: Returns all rows in x that aren't in y, where x and y are data frames.

**See Also**

[market\\_data](#)  
[market\\_profitability](#)  
[market\\_growth](#)  
[market\\_safety](#)

**Examples**

```
market_payouts(companies_r3k16[1,], financials_r3k16)
```

---

market\_profitability *Collects profitability z-scores for companies*

---

### Description

Given a data frame of companies (names and tickers) and a data frame of financial statements, calculates GPOA, ROE, ROA, CFOA, GMAR, ACC and determines the z-score of overall profitability for each company based on the paper Quality Minus Junk (Asness et al.) in Appendix page A2.

### Usage

```
market_profitability(  
  companies = qmj::companies_r3k16,  
  financials = qmj::financials_r3k16  
)
```

### Arguments

companies	A data frame of company names and tickers. Requires a 'ticker' column. Defaults to provided companies data set.
financials	A data frame containing financial statements for every company. Defaults to provided financial data set.

### Value

Whatdata.frame of market profitability values

### See Also

[market\\_data](#)  
[market\\_growth](#)  
[market\\_safety](#)  
[market\\_payouts](#)

### Examples

```
market_profitability(companies_r3k16[1,], financials_r3k16)
```

---

market_safety	<i>Collects safety z-scores for companies</i>
---------------	---

---

## Description

Given a data frame of companies (names and tickers), a data frame of financial statements, and a data frame of daily price data, calculates BAB, IVOL, LEV, O, Z, and EVOL, and determines the z-score of overall safety for each company based on the paper Quality Minus Junk (Asness et al.) in Appendix page A2.

## Usage

```
market_safety(  
  companies = qmj::companies_r3k16,  
  financials = qmj::financials_r3k16,  
  prices = qmj::prices_r3k16  
)
```

## Arguments

companies	A data frame of company names and tickers.
financials	A data frame containing financial statements for every company.
prices	A data frame containing the daily market closing prices and returns.

## Value

data.frame of market safety values

## See Also

[market\\_data](#)  
[market\\_profitability](#)  
[market\\_growth](#)  
[market\\_payouts](#)

## Examples

```
# Takes more than 10 secs  
market_safety(companies_r3k16[companies_r3k16$ticker %in% c("AAPL"), ])
```

---

prices_r3k16	<i>A dataframe of price returns and closing prices for companies in the Russell 3000 Index</i>
--------------	--

---

### Description

Stores price returns and closing prices for the past two years (if available) for the Russell 3000 Index companies as well as the S&P 500 (uniquely taken from Yahoo finance), to serve as a benchmark. For a description of the Russell 3000 index, as well as why it was used for this package, see [companies\\_r3k16](#). Last updated 2016/01/06.

### Format

A data frame with roughly 1,500,000 rows and 4 variables

- ticker = Company ticker, of class "character".
- date = Date in format YYYY-MM-DD, of class "character".
- pret = Price returns, of class "numeric".
- close = Closing stock prices for the day, of class "numeric".

### Details

Prices is used to calculate the safety score of companies, and stores closing stock prices and price returns for every company in [companies\\_r3k16](#) for the past two years. Price data varies significantly among companies, and companies that do not return price data are not represented here. Price returns are also calculated using two adjacent days in the dataset, a timespan which may cover one day or several depending on the company and what day is being considered.

### Source

Google Finance, accessed through quantmod

### See Also

[companies\\_r3k16](#)

[financials\\_r3k16](#)

## Description

The **qmj** package calculates quality scores for the companies in the Russell 3000 Index based on the paper *Quality Minus Junk* by Clifford Asness, Andrea Frazzini, and Lasse Pedersen.

## Details

Quality is a scaled measure of a company's profitability, growth, safety, and payouts. By using publicly available data for company balance sheets, income statements, and cash flows, **qmj** calculates relative quality z-scores for companies.

All functions and datasets are documented, and are freely available for use. Index of datasets:

- `companies_r3k16` - A data frame of publicly traded companies in the Russell 3000 Index.
- `financials_r3k16` - Financial statements for companies in the `companies_r3k16` dataset.
- `prices_r3k16` - Daily prices and price returns for the past two years for each company.
- `quality_r3k16` - Measured quality z-scores and component scores

## Author(s)

**Maintainer:** Yanrong Song <yrsong129@gmail.com>

Authors:

- Anthony Tsou <anttsou@gmail.com>
- Eugene Choe <ec7@williams.edu>
- David Kane <dave.kane@gmail.com>
- Ryan Kwon <rynkwn@gmail.com>
- Zijie Zhu <zijie.miller.zhu@gmail.com>

## References

Asness, Clifford S., Andrea Frazzini, and Lasse H. Pedersen. 'Quality Minus Junk.' AQR (2013)

## See Also

Useful links:

- Report bugs at <https://github.com/anttsou/qmj/issues>

---

`quality_r3k16`*A dataframe of quality scores for companies listed in the Russell 3000*

---

### Description

Displays overall quality scores as well as the scores for profitability, growth, safety, and payouts. Companies are sorted in order of quality score, with NAs stored at the end of the data set. For a description of the Russell 3000 index, as well as why it was used for this package, see [companies\\_r3k16](#). Last updated 2016/01/06.

### Format

A data frame with approximately 3000 rows and 7 variables

- `quality` = class "numeric".
- `profitability` = class "numeric".
- `growth` = class "numeric".
- `safety` = class "numeric".
- `payouts` = class "numeric".

### Details

The quality data set stores quality and component scores for the various companies list in the [companies\\_r3k16](#) data set. For every ticker in `companies`, quality attempts to assign a profitability, growth, safety, and payouts score to each company using data from [financials\\_r3k16](#) and [prices\\_r3k16](#), and then attempts to provide a quality score. It is possible that one or more companies may not have sufficient information to provide one or more component scores, in which case those companies can still be found at the end of the data set, with NA's making up any data that cannot be found.

If partial information exists (i.e., a profitability score was able to be calculated), then those scores are kept for that company, even if insufficient information exists to produce a quality score. More details may be found on the technical vignette.

---

`tidyinfo`*Formats raw financial data.*

---

### Description

`tidyinfo` works by formatting and curtailing the raw data generated by `quantmod` (and, by extension, the `get_info` function of this package)

### Usage

```
tidyinfo(x)
```

**Arguments**

x                    A list of lists of financial statements. Generated from `get_info(companies)`.

**Value**

Returns a data set that is usable by the other functions of this package, as well as being generally more readable.

data.frame of cleaned info (cash flows, income statements, balance sheets)

**See Also**

[get\\_info](#)

[tidy\\_cashflows](#)

[tidy\\_balancesheets](#)

[tidy\\_incomestatements](#)

**Examples**

```
if (reticulate::py_module_available("yfinance")) {  
  my_companies <- data.frame(ticker = c('GOOG', 'IBM'))  
  raw_data <- get_info(my_companies)  
  financials <- tidyinfo(raw_data)  
}
```

---

`tidy_balancesheets`     *Makes raw balancesheet data usable and readable.*

---

**Description**

Processes raw balance sheet data produced from `quantmod` into a tidy data frame. Raw balance sheet data must be formatted in a list such that every element is a data frame or matrix containing `quantmod` data.

**Usage**

```
tidy_balancesheets(x)
```

**Arguments**

x                    A list of raw cash flow data produced from `quantmod`

**Details**

`tidy_balancesheets` produces a data frame that is 'tidy' or more readily readable by a user and usable by other functions within this package.

**Value**

Returns a data set that's been 'tidied' up for use by other functions in this package.

**See Also**

[get\\_info](#)

[tidyinfo](#)

[tidy\\_cashflows](#)

[tidy\\_incomestatements](#)

---

tidy\_cashflows

*Makes raw cash flow data usable and readable.*

---

**Description**

Processes raw cash flow data from quantmod to return a tidied data frame. Raw cash flow data must be formatted in a list such that every element is a data frame or matrix containing quantmod data.

**Usage**

```
tidy_cashflows(x)
```

**Arguments**

x                    A list of raw cash flow data produced from quantmod

**Details**

tidy\_cashflows produces a data frame that is 'tidy' or more readily readable by a user and usable by other functions within this package.

**Value**

Returns a data set that's been 'tidied' up for use by other functions in this package.

**See Also**

[get\\_info](#)

[tidyinfo](#)

[tidy\\_balancesheets](#)

[tidy\\_incomestatements](#)

---

tidy_helper	<i>Main helper function for all tidy functions.</i>
-------------	---

---

**Description**

This function does the main work of converting raw financial data into organized data frames. It is used by qmj's tidy functions to reuse common code and to avoid potential mistakes from repeating similar processes.

**Usage**

```
tidy_helper(x)
```

**Arguments**

x	A matrix containing financial information, either cash flows, balancesheets, or income statements, downloaded from Google Finance. The formatting of the matrix has not been altered yet, as if just retrieved.
---	---

**See Also**

[tidy\\_cashflows](#)  
[tidy\\_balancesheets](#)  
[tidy\\_incomestatements](#)

---

tidy_incomestatements	<i>Makes raw incomestatement data usable and readable.</i>
-----------------------	--

---

**Description**

Tidies raw income statement data produced from quantmod and returns the tidied data frame. Raw income statement data must be formatted in a list such that every element is a data frame or matrix containing quantmod data.

**Usage**

```
tidy_incomestatements(x)
```

**Arguments**

x	A list of raw incomestatement file data produced from quantmod
---	--

**Details**

tidy\_incomestatements produces a data frame that is 'tidy' or more readily readable by a user and usable by other functions within this package.

**Value**

Returns a data set that's been 'tidied' up for use by other functions in this package.

**See Also**

[get\\_info](#)

[tidyinfo](#)

[tidy\\_cashflows](#)

[tidy\\_balancesheets](#)

---

tidy\_prices

*Formats raw price data.*

---

**Description**

Tidies raw prices and returns a tidied, usable data frame. Raw data should be structured identically to that produced by `get_prices()`, as this function depends on that structure.

**Usage**

```
tidy_prices(x)
```

**Arguments**

x                      Raw daily data, as produced by `get_prices()`

**Details**

`tidy_prices` produces a data frame that is 'tidy' or more readily readable by a user and usable by other functions within this package.

**Value**

Returns a data set that's been 'tidied' up for use by other functions in this package.

data.frame of cleaned prices

**Functions**

- `tidy_prices()`: combines relevant columns from the original price data set. Convert certain columns into character in order to bypass the warning generated by `dplyr::bind_rows()`

**See Also**

[get\\_prices](#)

**Examples**

```
my_companies <- data.frame(ticker=c('GOOG', 'IBM'))  
raw_price_data <- get_prices(my_companies)  
prices <- tidy_prices(raw_price_data)
```

# Index

## \* data

- companies\_r3k16, [3](#)
- financials\_r3k16, [4](#)
- prices\_r3k16, [14](#)
- quality\_r3k16, [16](#)

clean\_downloads, [2](#), [7](#), [8](#)  
companies\_r3k16, [3](#), [4](#), [5](#), [14](#), [16](#)

financials\_r3k16, [4](#), [4](#), [14](#), [16](#)

get\_companies, [5](#)  
get\_info, [3](#), [6](#), [8](#), [17](#), [18](#), [20](#)  
get\_prices, [3](#), [7](#), [7](#), [20](#)

install\_yfinance, [8](#)

market\_data, [9](#), [10–13](#)  
market\_growth, [10](#), [10](#), [11–13](#)  
market\_payouts, [10](#), [11](#), [12](#), [13](#)  
market\_profitability, [10](#), [11](#), [12](#), [13](#)  
market\_safety, [10–12](#), [13](#)

prices\_r3k16, [4](#), [5](#), [14](#), [16](#)

qmj, [15](#)  
qmj-package (qmj), [15](#)  
quality\_r3k16, [16](#)

tidy\_balancesheets, [17](#), [17](#), [18–20](#)  
tidy\_cashflows, [17](#), [18](#), [18](#), [19](#), [20](#)  
tidy\_helper, [19](#)  
tidy\_incomestatements, [17–19](#), [19](#)  
tidy\_prices, [8](#), [9](#), [20](#)  
tidyinfo, [7](#), [9](#), [16](#), [18](#), [20](#)