Package 'pmev'

August 27, 2024

Title Calculates Earned Value for a Project Schedule

Version 0.1.2

Author David Hammond [aut, cre]

URL https://github.com/david-hammond/pmev

BugReports https://github.com/david-hammond/pmev/issues

Maintainer David Hammond <anotherdavidhammond@gmail.com>

Description Given a project schedule and associated costs, this package calculates the earned value to date. It is an implementation of Project Management Body of Knowledge (PMBOK) methodologies (reference Project Management Institute. (2021). A guide to the Project Management Body of Knowledge (PMBOK guide) (7th ed.). Project Management Institute, Newtown Square, PA, ISBN 9781628256673 (pdf)).

License MIT + file LICENSE

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.2

Imports dplyr, lubridate, zoo, rlang, ggplot2, scales, vdiffr

Depends R (>= 2.10)

LazyData true

NeedsCompilation no

Repository CRAN

Date/Publication 2024-08-27 13:30:02 UTC

Contents

pmev-package	*pmev: Calculate Earned Value from a Project Schedule and Associated Costs.*

Description

pmev implements project management tracking metrics as outlined in the Project Management Body of Knowledge (PMBOK) manual which can be found here.

Details

From an inputted list of project activities, start and end dates, planned costs, progress and costs to date, it calculates the following PMBOK metrics:

- **Planned Value (PV)** Planned Value is the amount of the task that is supposed to have been completed, in terms of the task budget. It is calculated from the project budget by:
  - PV = Percent Complete (planned) x Task Budget.
- **Earned Value (EV)** Earned Value is the amount of the task that is actually completed. It is also calculated from the project budget.
  - EV = Percent Complete (actual) x Task Budget.
- **Actual Cost (AC)** Actual Cost is the actual to-date cost of the task.
- **Schedule Variance (SV)** A value which tells you the amount that the task is ahead or behind schedule.
  - SV = EV – PV.
    - If SV is negative, the task is behind schedule.
    - If SV is zero, the task is on schedule
    - If SV is positive, the task is ahead of schedule.
- **Schedule Performance Index (SPI)** The SPI, similar to the SV, also indicates ahead or behind schedule but gives the project manager a sense of the relative amount of the variance.
  - SPI = EV / PV
    - If SPI < 1, the task is behind schedule
    - If SPI = 1, the task is on schedule
    - If SPI > 1, the task is ahead of schedule
- **Cost Variance (CV)** Cost Variance tells the project manager how far the task is over or under budget.
  - CV = EV – AC
    - If CV is negative, the task is over budget
    - If CV is zero, the project is on budget
    - If CV is positive, the project is under budget
- **Cost Performance Index (CPI)** The CPI, similar to the CV, also indicates over or under budget but gives the project manager a sense of the relative amount of the variance.

  - CPI = EV / AC
    * If CPI < 1, the task is over budget
    * If CPI = 1, the task is on budget
    * If CPI > 1, the task is under budget
- **Budget at Completion (BAC)** It is simply the total project budget, which is the aggregate of all of the task budgets.
- **Estimate at Completion (EAC)** This value tells the project manager what the overall project budget will be if everything else went according to plan.
  - `pmev` calculates this as EAC = BAC / CPI.
- **Estimate to Complete (ETC)** This value tells the project manager how much money must be spent from this point forward, to complete the project.
  - ETC = EAC – AC
- **Variance at Completion (VAC)** This value tells the project manager the forecasted cost variance (CV) at the completion of the project.
  - VAC = BAC – EAC
- **To Complete Performance Index (TCPI)** This value tells the project manager what CPI would be necessary to finish the project on budget. It gives an indication of how much efficiency needs to be found in the remainder of the project to make up for past negative variances.
  - `pmev` calculates this as TCPI = (BAC – EV) / (BAC – AC)

Author(s)

Maintainer: David Hammond <anotherdavidhammond@gmail.com>

See Also

Useful links:

- https://github.com/david-hammond/pmev
- Report bugs at https://github.com/david-hammond/pmev

`earned_value` *Calculate the Earned Value of a Project Schedule to Date*

Description

Given a set of project activities start times, end times, progress and costs, this function calculates the Earned Value at a certain Date

Usage

```
earned_value(
  start,
  end,
  progress,
  planned_cost,
  project_value,
  cost_to_date,
  date = today()
)
```

Arguments

start	Start Date of activity
end	End Date of activity
progress	Proportion between 0 and 1 representing percentage completed for each activity (1 = 100% complete)
planned_cost	The planned costs of each activity
project_value	The total value of the project
cost_to_date	The total amount spent on the project to date
date	Character date "YYYY-MM-DD". Defaults to today.

Value

A list of two data frames:

- **pv** Planned Value Schedule, a data frame with two columns:
  - **date:** Daily Dates over Project Schedule
  - **planned_value:** The Planned Value to be delivered at that date.
- **ev** Earned Value Calculations, a data frame with 15 columns:
  - **date:** Date of calculation
  - **total_value:** Total Value of the Project.
  - **budget_at_completion:** Aggregate costs of all of the task budgets
  - **project_complete:** Project Complete based on Earned Value and total Project Value.
  - **schedule_complete:** The difference in Earned Value and Planned value as a proportion of the Total Value.
  - **planned_value:** The amount of the project that is supposed to have been completed.
  - **earned_value:** The amount of the project that is actually completed
  - **actual_cost:** Actual Cost is the actual to-date cost of the project.
  - **schedule_variance:** The amount that the project is ahead or behind schedule.
  - **cost_variance:** How far the task is over or under budget.
  - **cost_performance_index:** Relative amount of the variance to Planned Value.
  - **estimate_at_completion:** What the overall project budget will be if everything else went according to plan.

- **estimate_to_complete:** How much money must be spent from this point forward, to complete the project.
- **variance_at_completion:** The forecasted cost variance (CV) at the completion of the project.
- **to_complete_performance_index:** What CPI would be necessary to finish the project on budget.

Examples

```
data(project)
earned_value(start = project$start,
             end = project$end,
             progress = project$progress,
             planned_cost = project$planned_cost,
             project_value = 150000,
             cost_to_date = 10000,
             date = "2024-07-03")
```

---

ev *Get linearly scaled variable based on optimal bins*

Description

Get linearly scaled variable based on optimal bins

Get linearly scaled variable based on optimal bins

Details

Finds outliers and then bands between 0 and 1 on optimal bins of non-outlier data

Public fields

`planned_value` (`numeric()`)
Planned value schedule

`earned_value` (`numeric()`)
Earned value calculations

Methods

Public methods:

- `ev$new()`
- `ev$plot()`
- `ev$clone()`

Method `new()`:

Usage:

```
ev$new(
  start,
  end,
  progress,
  planned_cost,
  project_value,
  cost_to_date,
  date = today()
)
```

Arguments:

`start` Start Date of activity

`end` End Date of activity

`progress` Proportion between 0 and 1 representing percentage completed for each activity (1 = 100% complete)

`planned_cost` The planned costs of each activity

`project_value` The total value of the project

`cost_to_date` The total amount spent on the project to date

`date` Character date "YYYY-MM-DD". Defaults to today.

Returns: A new ev object.

Method `plot()`**:** Plots the planned and earned values

Usage:

```
ev$plot()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
ev$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
data(project)
tmp <- ev$new(start = project$start,
                      end = project$end,
                      progress = project$progress,
                      planned_cost = project$planned_cost,
                      project_value = 150000,
                      cost_to_date = 10000,
                      date = "2024-07-03")
plot(tmp)
```

`project` *Dummy Project Schedule*

Description

A dataset of a dummy project. The variables are as follows:

Usage

```
data(project)
```

Format

A data frame with 12 rows and 4 variables

- **start:** Start Date of the activity.
- **end:** End Date of the activity.
- **progress:** Proportion between 0 and 1 representing percentage completed for each activity (1 = 100% complete)
- **planned_cost:** The planned costs of each activity

Index