Package 'finnts'

July 22, 2025

Title Microsoft Finance Time Series Forecasting Framework

Version 0.5.0

Description
Automated time series forecasting developed by Microsoft Finance. The Microsoft Finance Time Series Forecasting Framework, aka Finn, can be used to forecast any component of the income statement, balance sheet, or any other area of interest by finance. Any numerical quantity over time,
Finn can be used to forecast it. While it can be applied outside of the finance domain, Finn was built
to meet the needs of financial analysts to better forecast their businesses within a company, and has a lot of built in features that are specific to the needs of financial forecasters. Happy forecasting!

URL https://microsoft.github.io/finnts/,
https://github.com/microsoft/finnts

BugReports https://github.com/microsoft/finnts/issues

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports cli, Cubist, dials, digest, doParallel, dplyr, earth, feasts, foreach, fs, generics, glue, glmnet, gtools, hts, kernlab, lubridate, magrittr, methods, parallel, parsnip, plyr, purrr, recipes, rlang, rsample, rules, snakecase, stringr, tibble, tidyr, tidyselect, timetk, tune, vroom, workflows

Suggests arrow (>= 8.0.0), AzureStor, Boruta, corrr, knitr, Microsoft365R, notebookutils, qs, reactable, rmarkdown, sparklyr, testthat (>= 3.0.0), vip

Config/testthat/edition 3

Depends R (>= 4.0), modeltime

VignetteBuilder knitr

NeedsCompilation no

Author Mike Tokic [aut, cre] (ORCID: <https://orcid.org/0000-0002-7630-7055>), Aadharsh Kannan [aut] (ORCID: <https://orcid.org/0000-0002-6475-8211>)

Maintainer Mike Tokic <mftokic@gmail.com>

Repository CRAN

Date/Publication 2024-10-25 17:50:02 UTC

Contents

`ensemble_models` *Ensemble Models*

Description

Create ensemble model forecasts

Usage

```
ensemble_models(
  run_info,
  parallel_processing = NULL,
  inner_parallel = FALSE,
  num_cores = NULL,
  seed = 123
)
```

Arguments

`run_info`	run info using the `set_run_info()` function
`parallel_processing`	Default of NULL runs no parallel processing and forecasts each individual time series one after another. 'local_machine' leverages all cores on current machine Finn is running on. 'spark' runs time series in parallel on a spark cluster in Azure Databricks or Azure Synapse.

inner_parallel	Run components of forecast process inside a specific time series in parallel. Can only be used if parallel_processing is set to NULL or 'spark'.
num_cores	Number of cores to run when parallel processing is set up. Used when running parallel computations on local machine or within Azure. Default of NULL uses total amount of cores on machine minus one. Can't be greater than number of cores on machine minus 1.
seed	Set seed for random number generator. Numeric value.

Value

Ensemble model outputs are written to disk

Examples

```
data_tbl <- timetk::m4_monthly %>%
  dplyr::rename(Date = date) %>%
  dplyr::mutate(id = as.character(id)) %>%
  dplyr::filter(
    Date >= "2013-01-01",
    Date <= "2015-06-01",
    id == "M750"
  )

run_info <- set_run_info()

prep_data(run_info,
  input_data = data_tbl,
  combo_variables = c("id"),
  target_variable = "value",
  date_type = "month",
  forecast_horizon = 3
)

prep_models(run_info,
  models_to_run = c("arima", "glmnet"),
  num_hyperparameters = 2
)

train_models(run_info,
  run_global_models = FALSE
)

ensemble_models(run_info)
```

---

final_models *Final Models*

Description

Select Best Models and Prep Final Outputs

Usage

```
final_models(
  run_info,
  average_models = TRUE,
  max_model_average = 3,
  weekly_to_daily = TRUE,
  parallel_processing = NULL,
  inner_parallel = FALSE,
  num_cores = NULL
)
```

Arguments

run_info	run info using the set_run_info() function.
average_models	If TRUE, create simple averages of individual models and save the most accurate one.
max_model_average	Max number of models to average together. Will create model averages for 2 models up until input value or max number of models ran.
weekly_to_daily	If TRUE, convert a week forecast down to day by evenly splitting across each day of week. Helps when aggregating up to higher temporal levels like month or quarter.
parallel_processing	Default of NULL runs no parallel processing and forecasts each individual time series one after another. 'local_machine' leverages all cores on current machine Finn is running on. 'spark' runs time series in parallel on a spark cluster in Azure Databricks or Azure Synapse.
inner_parallel	Run components of forecast process inside a specific time series in parallel. Can only be used if parallel_processing is set to NULL or 'spark'.
num_cores	Number of cores to run when parallel processing is set up. Used when running parallel computations on local machine or within Azure. Default of NULL uses total amount of cores on machine minus one. Can't be greater than number of cores on machine minus 1.

Value

Final model outputs are written to disk.

Examples

```
data_tbl <- timetk::m4_monthly %>%
  dplyr::rename(Date = date) %>%
  dplyr::mutate(id = as.character(id)) %>%
```

```
  dplyr::filter(
    Date >= "2013-01-01",
    Date <= "2015-06-01"
  )

run_info <- set_run_info()

prep_data(run_info,
  input_data = data_tbl,
  combo_variables = c("id"),
  target_variable = "value",
  date_type = "month",
  forecast_horizon = 3
)

prep_models(run_info,
  models_to_run = c("arima", "ets"),
  back_test_scenarios = 3
)

train_models(run_info,
  run_global_models = FALSE
)

final_models(run_info)
```

`forecast_time_series` *Finn Forecast Framework*

Description

Calls the Finn forecast framework to automatically forecast any historical time series.

Usage

```
forecast_time_series(
  run_info = NULL,
  input_data,
  combo_variables,
  target_variable,
  date_type,
  forecast_horizon,
  external_regressors = NULL,
  hist_start_date = NULL,
  hist_end_date = NULL,
  combo_cleanup_date = NULL,
  fiscal_year_start = 1,
  clean_missing_values = TRUE,
```

```
  clean_outliers = FALSE,
  back_test_scenarios = NULL,
  back_test_spacing = NULL,
  modeling_approach = "accuracy",
  forecast_approach = "bottoms_up",
  parallel_processing = NULL,
  inner_parallel = FALSE,
  num_cores = NULL,
  target_log_transformation = FALSE,
  negative_forecast = FALSE,
  fourier_periods = NULL,
  lag_periods = NULL,
  rolling_window_periods = NULL,
  recipes_to_run = NULL,
  pca = NULL,
  models_to_run = NULL,
  models_not_to_run = NULL,
  run_global_models = NULL,
  run_local_models = TRUE,
  run_ensemble_models = NULL,
  average_models = TRUE,
  max_model_average = 3,
  feature_selection = FALSE,
  weekly_to_daily = TRUE,
  seed = 123,
  run_model_parallel = FALSE,
  return_data = TRUE,
  run_name = "finnts_forecast"
)
```

Arguments

run_info	Run info using set_run_info()
input_data	A data frame or tibble of historical time series data. Can also include external regressors for both historical and future data.
combo_variables	List of column headers within input data to be used to separate individual time series.
target_variable	The column header formatted as a character value within input data you want to forecast.
date_type	The date granularity of the input data. Finn accepts the following as a character string day, week, month, quarter, year.
forecast_horizon	Number of periods to forecast into the future.
external_regressors	List of column headers within input data to be used as features in multivariate models.

hist_start_date
: Date value of when your input_data starts. Default of NULL is to use earliest date value in input_data.

hist_end_date
: Date value of when your input_data ends.Default of NULL is to use the latest date value in input_data.

combo_cleanup_date
: Date value to remove individual time series that don't contain non-zero values after that specified date. Default of NULL is to not remove any time series and attempt to forecast all of them.

fiscal_year_start
: Month number of start of fiscal year of input data, aids in building out date features. Formatted as a numeric value. Default of 1 assumes fiscal year starts in January.

clean_missing_values
: If TRUE, cleans missing values. Only impute values for missing data within an existing series, and does not add new values onto the beginning or end, but does provide a value of 0 for said values. Turned off when running hierarchical forecasts.

clean_outliers
: If TRUE, outliers are cleaned and inputted with values more in line with historical data

back_test_scenarios
: Number of specific back test folds to run when determining the best model. Default of NULL will automatically choose the number of back tests to run based on historical data size, which tries to always use a minimum of 80% of the data when training a model.

back_test_spacing
: Number of periods to move back for each back test scenario. Default of NULL moves back 1 period at a time for year, quarter, and month data. Moves back 4 for week and 7 for day data.

modeling_approach
: How Finn should approach your data. Current default and only option is 'accuracy'. In the future this could evolve to other areas like optimizing for interpretability over accuracy.

forecast_approach
: How the forecast is created. The default of 'bottoms_up' trains models for each individual time series. 'grouped_hierarchy' creates a grouped time series to forecast at while 'standard_hierarchy' creates a more traditional hierarchical time series to forecast, both based on the hts package.

parallel_processing
: Default of NULL runs no parallel processing and forecasts each individual time series one after another. 'local_machine' leverages all cores on current machine Finn is running on. 'spark' runs time series in parallel on a spark cluster in Azure Databricks or Azure Synapse.

inner_parallel
: Run components of forecast process inside a specific time series in parallel. Can only be used if parallel_processing is set to NULL or 'spark'.

num_cores Number of cores to run when parallel processing is set up. Used when running parallel computations on local machine or within Azure. Default of NULL uses total amount of cores on machine minus one. Can't be greater than number of cores on machine minus 1.

target_log_transformation
: If TRUE, log transform target variable before training models.

negative_forecast
: If TRUE, allow forecasts to dip below zero.

fourier_periods
: List of values to use in creating fourier series as features. Default of NULL automatically chooses these values based on the date_type.

lag_periods List of values to use in creating lag features. Default of NULL automatically chooses these values based on date_type.

rolling_window_periods
: List of values to use in creating rolling window features. Default of NULL automatically chooses these values based on date type.

recipes_to_run List of recipes to run on multivariate models that can run different recipes. A value of NULL runs all recipes, but only runs the R1 recipe for weekly and daily date types, and also for global models to prevent memory issues. A value of "all" runs all recipes, regardless of date type or if it's a local/global model. A list like c("R1") or c("R2") would only run models with the R1 or R2 recipe.

pca If TRUE, run principle component analysis on any lagged features to speed up model run time. Default of NULL runs PCA on day and week date types across all local multivariate models, and also for global models across all date types.

models_to_run List of models to run. Default of NULL runs all models.

models_not_to_run
: List of models not to run, overrides values in models_to_run. Default of NULL doesn't turn off any model.

run_global_models
: If TRUE, run multivariate models on the entire data set (across all time series) as a global model. Can be override by models_not_to_run. Default of NULL runs global models for all date types except week and day.

run_local_models
: If TRUE, run models by individual time series as local models.

run_ensemble_models
: If TRUE, run ensemble models. Default of NULL runs ensemble models only for quarter and month date types.

average_models If TRUE, create simple averages of individual models.

max_model_average
: Max number of models to average together. Will create model averages for 2 models up until input value or max number of models ran.

feature_selection
: Implement feature selection before model training

weekly_to_daily
: If TRUE, convert a week forecast down to day by evenly splitting across each day of week. Helps when aggregating up to higher temporal levels like month or quarter.

seed	Set seed for random number generator. Numeric value.
run_model_parallel	If TRUE, runs model training in parallel, only works when parallel_processing is set to 'local_machine' or 'spark'. Recommended to use a value of FALSE and leverage inner_parallel for new features.
return_data	If TRUE, return the forecast results. Used to be backwards compatible with previous finnts versions. Recommended to use a value of FALSE and leverage get_forecast_data() for new features.
run_name	Name used when submitting jobs to external compute like Azure Batch. Formatted as a character string.

Value

A list of three separate data sets: the future forecast, the back test results, and the best model per time series.

Examples

```
run_info <- set_run_info()

finn_forecast <- forecast_time_series(
  run_info = run_info,
  input_data = m750 %>% dplyr::rename(Date = date),
  combo_variables = c("id"),
  target_variable = "value",
  date_type = "month",
  forecast_horizon = 3,
  back_test_scenarios = 6,
  run_model_parallel = FALSE,
  models_to_run = c("arima", "ets", "snaive"),
  return_data = FALSE
)

fcst_tbl <- get_forecast_data(run_info)

models_tbl <- get_trained_models(run_info)
```

---

get_forecast_data *Get Final Forecast Data*

Description

Get Final Forecast Data

Usage

```
get_forecast_data(run_info, return_type = "df")
```

Arguments

run_info	run info using the set_run_info() function
return_type	return type

Value

table of final forecast results

Examples

```
data_tbl <- timetk::m4_monthly %>%
  dplyr::rename(Date = date) %>%
  dplyr::mutate(id = as.character(id)) %>%
  dplyr::filter(
    id == "M2",
    Date >= "2012-01-01",
    Date <= "2015-06-01"
  )

run_info <- set_run_info()

prep_data(run_info,
  input_data = data_tbl,
  combo_variables = c("id"),
  target_variable = "value",
  date_type = "month",
  forecast_horizon = 3,
  recipes_to_run = "R1"
)

prep_models(run_info,
  models_to_run = c("arima", "ets"),
  num_hyperparameters = 1
)

train_models(run_info,
  run_local_models = TRUE
)

final_models(run_info,
  average_models = FALSE
)

fcst_tbl <- get_forecast_data(run_info)
```

---

get_prepped_data *Get Prepped Data*

Description

Get Prepped Data

Usage

```
get_prepped_data(run_info, recipe, return_type = "df")
```

Arguments

run_info	run info using the set_run_info() function
recipe	recipe to return. Either a value of "R1" or "R2"
return_type	return type

Value

table of prepped data

Examples

```
data_tbl <- timetk::m4_monthly %>%
  dplyr::rename(Date = date) %>%
  dplyr::mutate(id = as.character(id)) %>%
  dplyr::filter(
    id == "M2",
    Date >= "2012-01-01",
    Date <= "2015-06-01"
  )

run_info <- set_run_info()

prep_data(run_info,
  input_data = data_tbl,
  combo_variables = c("id"),
  target_variable = "value",
  date_type = "month",
  forecast_horizon = 3,
  recipes_to_run = "R1"
)

R1_prepped_data_tbl <- get_prepped_data(run_info,
  recipe = "R1"
)
```

get_prepped_models *Get Prepped Model Info*

Description

Get Prepped Model Info

Usage

```
get_prepped_models(run_info)
```

Arguments

run_info	run info using the set_run_info() function

Value

table with data related to model workflows, hyperparameters, and back testing

Examples

```
data_tbl <- timetk::m4_monthly %>%
  dplyr::rename(Date = date) %>%
  dplyr::mutate(id = as.character(id)) %>%
  dplyr::filter(
    id == "M2",
    Date >= "2012-01-01",
    Date <= "2015-06-01"
  )

run_info <- set_run_info()

prep_data(run_info,
  input_data = data_tbl,
  combo_variables = c("id"),
  target_variable = "value",
  date_type = "month",
  forecast_horizon = 3,
  recipes_to_run = "R1"
)

prep_models(run_info,
  models_to_run = c("arima", "ets"),
  num_hyperparameters = 1
)

prepped_models_tbl <- get_prepped_models(run_info = run_info)
```

`get_run_info` *Get run info*

Description

Lets you get all of the logging associated with a specific experiment or run.

Usage

```
get_run_info(
  experiment_name = NULL,
  run_name = NULL,
  storage_object = NULL,
  path = NULL
)
```

Arguments

`experiment_name`	Name used to group similar runs under a single experiment name.
`run_name`	Name to distinguish one run of Finn from another. The current time in UTC is appended to the run name to ensure a unique run name is created.
`storage_object`	Used to store outputs during a run to other storage services in Azure. Could be a storage container object from the 'AzureStor' package to connect to ADLS blob storage or a OneDrive/SharePoint object from the 'Microsoft365R' package to connect to a OneDrive folder or SharePoint site. Default of NULL will save outputs to the local file system.
`path`	String showing what file path the outputs should be written to. Default of NULL will write the outputs to a temporary directory within R, which will delete itself after the R session closes.

Value

Data frame of run log information

Examples

```
run_info <- set_run_info(
  experiment_name = "finn_forecast",
  run_name = "test_run"
)

run_info_tbl <- get_run_info(
  experiment_name = "finn_forecast"
)
```

`get_trained_models` *Get Final Trained Models*

Description

Get Final Trained Models

Usage

```
get_trained_models(run_info)
```

Arguments

`run_info`	run info using the `set_run_info()` function

Value

table of final trained models

Examples

```
data_tbl <- timetk::m4_monthly %>%
  dplyr::rename(Date = date) %>%
  dplyr::mutate(id = as.character(id)) %>%
  dplyr::filter(
    id == "M2",
    Date >= "2012-01-01",
    Date <= "2015-06-01"
  )

run_info <- set_run_info()

prep_data(run_info,
  input_data = data_tbl,
  combo_variables = c("id"),
  target_variable = "value",
  date_type = "month",
  forecast_horizon = 3,
  recipes_to_run = "R1"
)

prep_models(run_info,
  models_to_run = c("arima", "ets"),
  num_hyperparameters = 1
)

train_models(run_info,
  run_global_models = FALSE,
  run_local_models = TRUE
)
```

```
final_models(run_info,
  average_models = FALSE
)

models_tbl <- get_trained_models(run_info)
```

`list_models` *List all available models*

Description

List all available models

Usage

```
list_models()
```

Value

list of models

`prep_data` *Prep Data*

Description

Preps data with various feature engineering recipes to create features before training models

Usage

```
prep_data(
  run_info,
  input_data,
  combo_variables,
  target_variable,
  date_type,
  forecast_horizon,
  external_regressors = NULL,
  hist_start_date = NULL,
  hist_end_date = NULL,
  combo_cleanup_date = NULL,
  fiscal_year_start = 1,
  clean_missing_values = TRUE,
  clean_outliers = FALSE,
```

```
  box_cox = FALSE,
  stationary = TRUE,
  forecast_approach = "bottoms_up",
  parallel_processing = NULL,
  num_cores = NULL,
  target_log_transformation = FALSE,
  fourier_periods = NULL,
  lag_periods = NULL,
  rolling_window_periods = NULL,
  recipes_to_run = NULL,
  multistep_horizon = FALSE
)
```

Arguments

run_info	Run info using set_run_info()
input_data	A standard data frame, tibble, or spark data frame using sparklyr of historical time series data. Can also include external regressors for both historical and future data.
combo_variables	List of column headers within input data to be used to separate individual time series.
target_variable	The column header formatted as a character value within input data you want to forecast.
date_type	The date granularity of the input data. Finn accepts the following as a character string: day, week, month, quarter, year.
forecast_horizon	Number of periods to forecast into the future.
external_regressors	List of column headers within input data to be used as features in multivariate models.
hist_start_date	Date value of when your input_data starts. Default of NULL uses earliest date value in input_data.
hist_end_date	Date value of when your input_data ends. Default of NULL uses the latest date value in input_data.
combo_cleanup_date	Date value to remove individual time series that don't contain non-zero values after that specified date. Default of NULL is to not remove any time series and attempt to forecast all time series.
fiscal_year_start	Month number of start of fiscal year of input data, aids in building out date features. Formatted as a numeric value. Default of 1 assumes fiscal year starts in January.

`clean_missing_values`
: If TRUE, cleans missing values. Only impute values for missing data within an existing series, and does not add new values onto the beginning or end, but does provide a value of 0 for said values.

`clean_outliers`
: If TRUE, outliers are cleaned and inputted with values more in line with historical data.

`box_cox`
: Apply box-cox transformation to normalize variance in data

`stationary`
: Apply differencing to make data stationary

`forecast_approach`
: How the forecast is created. The default of 'bottoms_up' trains models for each individual time series. Value of 'grouped_hierarchy' creates a grouped time series to forecast at while 'standard_hierarchy' creates a more traditional hierarchical time series to forecast, both based on the hts package.

`parallel_processing`
: Default of NULL runs no parallel processing and forecasts each individual time series one after another. Value of 'local_machine' leverages all cores on current machine Finn is running on. Value of 'spark' runs time series in parallel on a spark cluster in Azure Databricks/Synapse.

`num_cores`
: Number of cores to run when parallel processing is set up. Used when running parallel computations on local machine or within Azure. Default of NULL uses total amount of cores on machine minus one. Can't be greater than number of cores on machine minus 1.

`target_log_transformation`
: If TRUE, log transform target variable before training models.

`fourier_periods`
: List of values to use in creating fourier series as features. Default of NULL automatically chooses these values based on the date_type.

`lag_periods`
: List of values to use in creating lag features. Default of NULL automatically chooses these values based on date_type.

`rolling_window_periods`
: List of values to use in creating rolling window features. Default of NULL automatically chooses these values based on date_type.

`recipes_to_run`
: List of recipes to run on multivariate models that can run different recipes. A value of NULL runs all recipes, but only runs the R1 recipe for weekly and daily date types. A value of "all" runs all recipes, regardless of date type. A list like c("R1") or c("R2") would only run models with the R1 or R2 recipe.

`multistep_horizon`
: Use a multistep horizon approach when training multivariate models with R1 recipe.

Value

No return object. Feature engineered data is written to disk based on the output locations provided in `set_run_info()`.

Examples

```
data_tbl <- timetk::m4_monthly %>%
  dplyr::rename(Date = date) %>%
  dplyr::mutate(id = as.character(id)) %>%
  dplyr::filter(
    Date >= "2013-01-01",
    Date <= "2015-06-01"
  )

run_info <- set_run_info()

prep_data(run_info,
  input_data = data_tbl,
  combo_variables = c("id"),
  target_variable = "value",
  date_type = "month",
  forecast_horizon = 3,
  recipes_to_run = "R1"
)
```

`prep_models` *Prep Models*

Description

Preps various aspects of run before training models. Things like train/test splits, creating hyperparameters, etc.

Usage

```
prep_models(
  run_info,
  back_test_scenarios = NULL,
  back_test_spacing = NULL,
  models_to_run = NULL,
  models_not_to_run = NULL,
  run_ensemble_models = TRUE,
  pca = NULL,
  num_hyperparameters = 10,
  seed = 123
)
```

Arguments

`run_info`	run info using the `set_run_info()` function.

back_test_scenarios
: Number of specific back test folds to run when determining the best model. Default of NULL will automatically choose the number of back tests to run based on historical data size, which tries to always use a minimum of 80% of the data when training a model.

back_test_spacing
: Number of periods to move back for each back test scenario. Default of NULL moves back 1 period at a time for year, quarter, and month data. Moves back 4 for week and 7 for day data.

models_to_run
: List of models to run. Default of NULL runs all models.

models_not_to_run
: List of models not to run, overrides values in models_to_run. Default of NULL doesn't turn off any model.

run_ensemble_models
: If TRUE, prep for ensemble models.

pca
: If TRUE, run principle component analysis on any lagged features to speed up model run time. Default of NULL runs PCA on day and week date types across all local multivariate models, and also for global models across all date types.

num_hyperparameters
: number of hyperparameter combinations to test out on validation data for model tuning.

seed
: Set seed for random number generator. Numeric value.

Value

Writes outputs related to model prep to disk.

Examples

```
data_tbl <- timetk::m4_monthly %>%
  dplyr::rename(Date = date) %>%
  dplyr::mutate(id = as.character(id)) %>%
  dplyr::filter(
    Date >= "2012-01-01",
    Date <= "2015-06-01"
  )

run_info <- set_run_info()

prep_data(run_info,
  input_data = data_tbl,
  combo_variables = c("id"),
  target_variable = "value",
  date_type = "month",
  forecast_horizon = 3
)

prep_models(run_info,
  models_to_run = c("arima", "ets", "glmnet")
```

```
)
```

`set_run_info` *Set up finnts submission*

Description

Creates list object of information helpful in logging information about your run.

Usage

```
set_run_info(
  experiment_name = "finn_fcst",
  run_name = "finn_fcst",
  storage_object = NULL,
  path = NULL,
  data_output = "csv",
  object_output = "rds",
  add_unique_id = TRUE
)
```

Arguments

`experiment_name`	Name used to group similar runs under a single experiment name.
`run_name`	Name to distinguish one run of Finn from another. The current time in UTC is appended to the run name to ensure a unique run name is created.
`storage_object`	Used to store outputs during a run to other storage services in Azure. Could be a storage container object from the 'AzureStor' package to connect to ADLS blob storage or a OneDrive/SharePoint object from the 'Microsoft365R' package to connect to a OneDrive folder or SharePoint site. Default of NULL will save outputs to the local file system.
`path`	String showing what file path the outputs should be written to. Default of NULL will write the outputs to a temporary directory within R, which will delete itself after the R session closes.
`data_output`	String value describing the file type for data outputs. Default will write data frame outputs as csv files. The other option of 'parquet' will instead write parquet files.
`object_output`	String value describing the file type for object outputs. Default will write object outputs like trained models as rds files. The other option of 'qs' will instead serialize R objects as qs files by using the 'qs' package.
`add_unique_id`	Add a unique id to end of run_name based on submission time. Set to FALSE to supply your own unique run name, which is helpful in multistage ML pipelines.

Value

A list of run information

Examples

```
run_info <- set_run_info(
  experiment_name = "test_exp",
  run_name = "test_run_1"
)
```

`train_models` *Train Individual Models*

Description

Train Individual Models

Usage

```
train_models(
  run_info,
  run_global_models = FALSE,
  run_local_models = TRUE,
  global_model_recipes = c("R1"),
  feature_selection = FALSE,
  negative_forecast = FALSE,
  parallel_processing = NULL,
  inner_parallel = FALSE,
  num_cores = NULL,
  seed = 123
)
```

Arguments

`run_info`	run info using the `set_run_info()` function
`run_global_models`	If TRUE, run multivariate models on the entire data set (across all time series) as a global model. Can be override by models_not_to_run. Default of NULL runs global models for all date types except week and day.
`run_local_models`	If TRUE, run models by individual time series as local models.
`global_model_recipes`	Recipes to use in global models.
`feature_selection`	Implement feature selection before model training

negative_forecast
: If TRUE, allow forecasts to dip below zero.

parallel_processing
: Default of NULL runs no parallel processing and forecasts each individual time series one after another. 'local_machine' leverages all cores on current machine Finn is running on. 'spark' runs time series in parallel on a spark cluster in Azure Databricks or Azure Synapse.

inner_parallel
: Run components of forecast process inside a specific time series in parallel. Can only be used if parallel_processing is set to NULL or 'spark'.

num_cores
: Number of cores to run when parallel processing is set up. Used when running parallel computations on local machine or within Azure. Default of NULL uses total amount of cores on machine minus one. Can't be greater than number of cores on machine minus 1.

seed
: Set seed for random number generator. Numeric value.

Value

trained model outputs are written to disk.

Examples

```
data_tbl <- timetk::m4_monthly %>%
  dplyr::rename(Date = date) %>%
  dplyr::mutate(id = as.character(id)) %>%
  dplyr::filter(
    Date >= "2013-01-01",
    Date <= "2015-06-01"
  )

run_info <- set_run_info()

prep_data(run_info,
  input_data = data_tbl,
  combo_variables = c("id"),
  target_variable = "value",
  date_type = "month",
  forecast_horizon = 3
)

prep_models(run_info,
  models_to_run = c("arima", "glmnet"),
  num_hyperparameters = 2,
  back_test_scenarios = 6,
  run_ensemble_models = FALSE
)

train_models(run_info)
```

Index