hdhp.py Documentation

Release 1.0

Charalampos Mavroforakis and contributors

Dec 10, 2016

Contents:

CHAPTER 1

The `infer` function

`hdhp.`**`infer`**(*history*, *alpha_0*, *mu_0*, *omega=1*, *beta=1*, *theta_0=None*, *threads=1*, *num_particles=1*, *particle_weight_threshold=1*, *resample_every=10*, *update_kernels=True*, *mu_rate=0.6*, *keep_alpha_history=False*, *progress_file=None*, *seed=None*)

Runs the inference algorithm and returns a particle.

Parameters

- **`history`** (*`list`*) – A list of 4-tuples (user, time, content, metadata) that represents the event history that we want to infer our model on.
- **`alpha_0`** (*`tuple`*) – The Gamma prior parameter for a pattern's time kernel.
- **`mu_0`** (*`tuple`*) – The Gamma prior parameter for the user activity rate.
- **`omega`** (*`float`*) – The time decay parameter.
- **`beta`** (*`float`*) – A parameter that controls the new-task probability.
- **`theta_0`** (*`list, default is None`*) – If not None, theta_0 corresponds to the Dirichlet prior used for the word distribution. It should have as many dimensions as the number of words. By default, this is the vector $[1/|V|, \ldots, 1/|V|]$, where $|V|$ is the size of the vocabulary.
- **`threads`** (*`int, default is 1`*) – The number of CPU threads that will be used during inference.
- **`num_particles`** (*`int, default is 1`*) – The number of particles that the SMC algorithm will use.
- **`particle_weight_threshold`** (*`float, default is 1`*) – A parameter that controls when the particles need to be resampled
- **`resample_every`** (*`int, default is 10`*) – The frequency with which we check if we need to resample or not. The number is in inference steps (number of events)
- **`update_kernels`** (*`bool, default is True`*) – Controls wheter the time kernel parameter of each pattern will be updated from the posterior, or not.

- **mu_rate** (*float, default is 0.6*) – The learning-rate with which we update the activity rate of a user.
- **keep_alpha_history** (*bool, default is False*) – For debug reasons, we make want to keep the complete history of the value of each pattern's time kernel parameter as we see more events in that pattern.
- **progress_file** (*str, default is None*) – Since the computation might be slow, we want to save progress information to a file instead of printing it. If None, a temporary, randomly-named file is generated for this purpose.

CHAPTER 2

The `HDHProcess` object

class `hdhp.`**`HDHProcess`**(*num_patterns*, *alpha_0*, *mu_0*, *vocabulary*, *omega=1*, *doc_length=20*, *doc_min_length=5*, *words_per_pattern=10*, *random_state=None*)

`__init__`(*num_patterns*, *alpha_0*, *mu_0*, *vocabulary*, *omega=1*, *doc_length=20*, *doc_min_length=5*, *words_per_pattern=10*, *random_state=None*)

Parameters

- **`num_patterns`** (*`int`*) – The maximum number of patterns that will be shared across the users.
- **`alpha_0`** (*`tuple`*) – The parameter that is used when sampling the time kernel weights of each pattern. The distribution that is being used is a Gamma. This tuple should be of the form (shape, scale).
- **`mu_0`** (*`tuple`*) – The parameter of the Gamma distribution that is used to sample each user's mu (activity level). This tuple should be of the form (shape, scale).
- **`vocabulary`** (*`list`*) – The list of available words to use when generating documents.
- **`omega`** (*`float, default is 1`*) – The decay parameter for the decay of the exponential decay kernel.
- **`doc_length`** (*`int, default is 20`*) – The maximum number of words per document.
- **`doc_min_length`** (*`int, default is 5`*) – The minimum number of words per document.
- **`words_per_pattern`** (*`int, default is 10`*) – The number of words that will have a non-zero probability to appear in each pattern.
- **`random_state`** (*`int or RandomState object, default is None`*) – The random number generator.

`kernel`(*t_i*, *t_j*)

Returns the kernel function for t_i and t_j.

Parameters

- **t_i** (*float*) – Timestamp representing *now*.
- **t_j** (*float*) – Timestamp representaing *past*.

Returns

Return type float

pattern_content_str (*patterns=None*, *show_words=-1*, *min_word_occurence=5*)
Return the content information for the patterns of the process.

Parameters

- **patterns** (*list, default is None*) – If this list is provided, only information about the patterns in the list will be returned.
- **show_words** (*int, default is -1*) – The maximum number of words to show for each pattern. Notice that the words are sorted according to their occurence count.
- **min_word_occurence** (*int, default is 5*) – Only show words that show up at least *min_word_occurence* number of times in the documents of the respective pattern.

Returns A string with all the content information

Return type str

plot (*num_samples=500*, *T_min=0*, *T_max=None*, *start_date=None*, *users=None*, *user_limit=50*, *patterns=None*, *task_detail=False*, *save_to_file=False*, *filename='user_timelines'*, *intensity_threshold=None*, *paper=True*, *colors=None*, *fig_width=20*, *fig_height_per_user=5*, *time_unit='months'*, *label_every=3*, *seed=None*)
Plots the intensity of a set of users for a set of patterns over a time period.

In this plot, each user is a separate subplot and for each user the plot shows her event_rate for each separate pattern that she has been active at.

Parameters

- **num_samples** (*int, default is 500*) – The granularity level of the intensity line. Smaller number of samples results in faster plotting, while larger numbers give much more detailed result.
- **T_min** (*float, default is 0*) – The minimum timestamp that the plot shows, in seconds.
- **T_max** (*float, default is None*) – If not None, this is the maximum timestamp that the plot considers, in seconds.
- **start_date** (*datetime, default is None*) – If provided, this is the actual datetime that corresponds to time 0. This is required if *paper* is True.
- **users** (*list, default is None*) – If provided, this list contains the id's of the users that will be plotted. Actually, only the first *user_limit* of them will be shown.
- **user_limit** (*int, default is 50*) – The maximum number of users to plot.
- **patterns** (*list, default is None*) – The list of patterns that will be shown in the final plot. If None, all of the patterns will be plotted.
- **task_detail** (*bool, default is False*) – If True, thee plot has one line per task. Otherwise, we only plot the commulative intensity of all tasks under the same pattern.
- **save_to_file** (*bool, default is False*) – If True, the plot will be saved to a *pdf* and a *png* file.
- **filename** (*str, default is 'user_timelines'*) – The name of the output file that will be used when saving the plot.

- **`intensity_threshold`** (*`float, default is None`*) – If provided, this is the maximum intensity value that will be plotted, i.e. the y_max that will be the cut-off threshold for the y-axis.
- **`paper`** (*`bool, default is True`*) – If True, the plot result will be the same as the figures that are in the published paper.
- **`colors`** (*`list, default is None`*) – A list of colors that will be used for the plot. Each color will correspond to a single pattern, and will be shared across all the users.
- **`fig_width`** (*`int, default is 20`*) – The width of the figure that will be returned.
- **`fig_height_per_user`** (*`int, default is 5`*) – The height of each separate user-plot of the final figure. If multiplied by the number of users, this determines the total height of the figure. Notice that due to a matplotlib constraint(?) the total height of the figure cannot be over 70.
- **`time_unit`** (*`str, default is 'months'`*) – Controls wether the time units is measured in days (in which case it should be set to 'days') or months.
- **`label_every`** (*`int, default is 3`*) – The frequency of the labels that show in the x-axis.
- **`seed`** (*`int, default is None`*) – A seed to the random number generator used to assign colors to patterns.

Returns **fig**

Return type matplotlib.Figure object

`reset`()

Removes all the events and users already sampled.

Note: It does not reseed the random number generator. It also retains the already sampled pattern parameters (word distributions and alphas)

`sample_document`(*pattern*)

Sample a random document from a specific pattern.

Parameters **`pattern`** (*`int`*) – The pattern from which to sample the content.

Returns A space separeted string that contains all the sampled words.

Return type str

`sample_mu`()

Samples a value from the prior of the base intensity mu.

Returns **mu_u** – The base intensity of a user, sampled from the prior.

Return type float

`sample_next_time`(*pattern, user*)

Samples the time of the next event of a pattern for a given user.

Parameters

- **`pattern`** (*`int`*) – The pattern index that we want to sample the next event for.
- **`user`** (*`int`*) – The index of the user that we want to sample for.

Returns **timestamp**

Return type float

sample_pattern_params()

Returns the word distributions for each pattern.

Returns **parameters** – A list of word distributions, one for each pattern.

Return type list

sample_pattern_popularity()

Returns a popularity distribution over the patterns.

Returns **pattern_popularities** – A list with the popularity distribution of each pattern.

Return type list

sample_time_kernels()

Returns the time decay parameter of each pattern.

Returns **alphas** – A list of time decay parameters, one for each pattern.

Return type list

sample_user_events(*min_num_events=100*, *max_num_events=None*, *t_max=None*)

Samples events for a user.

Parameters

- **min_num_events** (*int, default is 100*) – The minimum number of events to sample.
- **max_num_events** (*int, default is None*) – If not None, this is the maximum number of events to sample.
- **t_max** (*float, default is None*) – The time limit until which to sample events.

Returns **events** – A list of the form [(t_i, doc_i, user_i, meta_i), ...] sorted by increasing time that has all the events of the sampled users. Above, doc_i is the document and meta_i is any sort of metadata that we want for doc_i, e.g. question_id. The generator will return an empty list for meta_i.

Return type list

show_annotated_events(*user=None*, *patterns=None*, *show_time=True*, *T_min=0*, *T_max=None*)

Returns a string where each event is annotated with the inferred pattern.

Parameters

- **user** (*int, default is None*) – If given, the events returned are limited to the selected user
- **patterns** (*list, default is None*) – If not None, an event is return only if it belongs to one of the selected patterns
- **show_time** (*bool, default is True*) – Controls whether the time of the event will be shown
- **T_min** (*float, default is 0*) – Controls the minimum timestamp after which the events will be shown
- **T_max** (*float, default is None*) – If given, T_max controls the maximum timestamp shown

Returns

Return type str

show_pattern_content (*patterns=None*, *words=0*, *detail_threshold=5*)

Shows the content distrubution of the inferred patterns.

Parameters

- **patterns** (*list, default is None*) – If not None, only the content of the selected patterns will be shown
- **words** (*int, default is 0*) – A positive number that control how many words will be shown. The words are being shown sorted by their likelihood, starting with the most probable.
- **detail_threshold** (*int, default is 5*) – A positive number that sets the lower bound in the number of times that a word appeared in a pattern so that its count is shown.

Returns

Return type str

user_pattern_history_str (*user=None*, *patterns=None*, *show_time=True*, *t_min=0*)

Returns a representation of the history of a user's actions and the pattern that they correspond to.

Parameters

- **user** (*int, default is None*) – An index to the user we want to inspect. If None, the function runs over all the users.
- **patterns** (*list, default is None*) – If not None, limit the actions returned to the ones that belong in the provided patterns.
- **show_time** (*bool, default is True*) – Control wether the timestamp will appear in the representation or not.
- **t_min** (*float, default is 0*) – The timestamp after which we only consider actions.

Returns

Return type str

user_patterns (*user*)

Returns a list with the patterns that a user has adopted.

Parameters **user** (*int*) –

user_patterns_set (*user*)

Return the patterns that a specific user adopted.

Parameters **user** (*int*) – The index of a user.

Returns The set of the patterns that the user adopted.

Return type set

Python Module Index

Index