`plotly.express` (plotly.express.html#module-plotly.express).histogram

`plotly.express.` **`histogram`**(*data_frame=None*, *x=None*, *y=None*, *color=None*, *pattern_shape=None*, *facet_row=None*, *facet_col=None*, *facet_col_wrap=0*, *facet_row_spacing=None*, *facet_col_spacing=None*, *hover_name=None*, *hover_data=None*, *animation_frame=None*, *animation_group=None*, *category_orders=None*, *labels=None*, *color_discrete_sequence=None*, *color_discrete_map=None*, *pattern_shape_sequence=None*, *pattern_shape_map=None*, *marginal=None*, *opacity=None*, *orientation=None*, *barmode='relative'*, *barnorm=None*, *histnorm=None*, *log_x=False*, *log_y=False*, *range_x=None*, *range_y=None*, *histfunc=None*, *cumulative=None*, *nbins=None*, *text_auto=False*, *title=None*, *template=None*, *width=None*, *height=None*) → plotly.graph_objects._figure.Figure

In a histogram, rows of `data_frame` are grouped together into a rectangular mark to visualize the 1D distribution of an aggregate function `histfunc` (e.g. the count or sum) of the value `y` (or `x` if `orientation` is `'h'`).

Parameters:

- **data_frame** (*DataFrame or array-like or dict* (https://docs.python.org/3/library/stdtypes.html#dict)) – This argument needs to be passed for column names (and not keyword names) to be used. Array-like and dict are transformed internally to a pandas DataFrame. Optional: if missing, a DataFrame gets constructed under the hood using the other arguments.
- **x** (*str* (https://docs.python.org/3/library/stdtypes.html#str) *or int* (https://docs.python.org/3/library/functions.html#int) *or Series or array-like*) – Either a name of a column in `data_frame`, or a pandas Series or array_like object. Values from this column or array_like are used to position marks along the x axis in cartesian coordinates. If `orientation` is `'h'`, these values are used as inputs to `histfunc`. Either `x` or `y` can optionally be a list of column references or array_likes, in which case the data will be treated as if it were ‘wide’ rather than ‘long’.
- **y** (*str* (https://docs.python.org/3/library/stdtypes.html#str) *or int* (https://docs.python.org/3/library/functions.html#int) *or Series or array-like*) – Either a name of a column in `data_frame`, or a pandas Series or array_like object. Values from this column or array_like are used to position marks along the y axis in cartesian coordinates. If `orientation` is `'v'`, these values are used as inputs to `histfunc`. Either `x` or `y` can optionally be a list of column references or array_likes, in which case the data will be treated as if it were ‘wide’ rather than ‘long’.

- **color** (*str* (https://docs.python.org/3/library/stdtypes.html#str) *or int* (https://docs.python.org/3/library/functions.html#int) *or Series or array-like*) – Either a name of a column in `data_frame`, or a pandas Series or array_like object. Values from this column or array_like are used to assign color to marks.
- **pattern_shape** (*str* (https://docs.python.org/3/library/stdtypes.html#str) *or int* (https://docs.python.org/3/library/functions.html#int) *or Series or array-like*) – Either a name of a column in `data_frame`, or a pandas Series or array_like object. Values from this column or array_like are used to assign pattern shapes to marks.
- **facet_row** (*str* (https://docs.python.org/3/library/stdtypes.html#str) *or int* (https://docs.python.org/3/library/functions.html#int) *or Series or array-like*) – Either a name of a column in `data_frame`, or a pandas Series or array_like object. Values from this column or array_like are used to assign marks to facetted subplots in the vertical direction.
- **facet_col** (*str* (https://docs.python.org/3/library/stdtypes.html#str) *or int* (https://docs.python.org/3/library/functions.html#int) *or Series or array-like*) – Either a name of a column in `data_frame`, or a pandas Series or array_like object. Values from this column or array_like are used to assign marks to facetted subplots in the horizontal direction.
- **facet_col_wrap** (*int* (https://docs.python.org/3/library/functions.html#int)) – Maximum number of facet columns. Wraps the column variable at this width, so that the column facets span multiple rows. Ignored if 0, and forced to 0 if `facet_row` or a `marginal` is set.
- **facet_row_spacing** (*float between 0 and 1*) – Spacing between facet rows, in paper units. Default is 0.03 or 0.07 when facet_col_wrap is used.
- **facet_col_spacing** (*float between 0 and 1*) – Spacing between facet columns, in paper units Default is 0.02.
- **hover_name** (*str* (https://docs.python.org/3/library/stdtypes.html#str) *or int* (https://docs.python.org/3/library/functions.html#int) *or Series or array-like*) – Either a name of a column in `data_frame`, or a pandas Series or array_like object. Values from this column or array_like appear in bold in the hover tooltip.
- **hover_data** (*str* (https://docs.python.org/3/library/stdtypes.html#str)*, or list of str or int* (https://docs.python.org/3/library/functions.html#int)*, or Series or array-like, or dict* (https://docs.python.org/3/library/stdtypes.html#dict)) – Either a name or list of names of columns in `data_frame`, or pandas Series, or array_like objects or a dict with column names as keys, with values True (for default formatting) False (in order to remove this column from hover information), or a formatting string, for example ':.3f' or '|%a' or list-like data to appear in the hover tooltip or tuples with a bool or formatting string as first element, and list-like data to appear in hover as second element Values from these columns appear as extra data in the hover tooltip.
- **animation_frame** (*str* (https://docs.python.org/3/library/stdtypes.html#str) *or int* (https://docs.python.org/3/library/functions.html#int) *or Series or array-like*) – Either a name of a column in `data_frame`, or a pandas Series or array_like object. Values

from this column or array_like are used to assign marks to animation frames.

- **animation_group** (*str* (https://docs.python.org/3/library/stdtypes.html#str) *or int* (https://docs.python.org/3/library/functions.html#int) *or Series or array-like*) – Either a name of a column in `data_frame`, or a pandas Series or array_like object. Values from this column or array_like are used to provide object-constancy across animation frames: rows with matching `animation_group`s will be treated as if they describe the same object in each frame.
- **category_orders** (dict with str keys and list of str values (default `{}`)) – By default, in Python 3.6+, the order of categorical values in axes, legends and facets depends on the order in which these values are first encountered in `data_frame` (and no order is guaranteed by default in Python below 3.6). This parameter is used to force a specific ordering of values per column. The keys of this dict should correspond to column names, and the values should be lists of strings corresponding to the specific display order desired.
- **labels** (dict with str keys and str values (default `{}`)) – By default, column names are used in the figure for axis titles, legend entries and hovers. This parameter allows this to be overridden. The keys of this dict should correspond to column names, and the values should correspond to the desired label to be displayed.
- **color_discrete_sequence** (*list of str*) – Strings should define valid CSS-colors. When `color` is set and the values in the corresponding column are not numeric, values in that column are assigned colors by cycling through `color_discrete_sequence` in the order described in `category_orders`, unless the value of `color` is a key in `color_discrete_map`. Various useful color sequences are available in the `plotly.express.colors` submodules, specifically `plotly.express.colors.qualitative`.
- **color_discrete_map** (dict with str keys and str values (default `{}`)) – String values should define valid CSS-colors Used to override `color_discrete_sequence` to assign a specific colors to marks corresponding with specific values. Keys in `color_discrete_map` should be values in the column denoted by `color`. Alternatively, if the values of `color` are valid colors, the string `'identity'` may be passed to cause them to be used directly.
- **pattern_shape_sequence** (*list of str*) – Strings should define valid plotly.js patterns-shapes. When `pattern_shape` is set, values in that column are assigned patterns-shapes by cycling through `pattern_shape_sequence` in the order described in `category_orders`, unless the value of `pattern_shape` is a key in `pattern_shape_map`.
- **pattern_shape_map** (dict with str keys and str values (default `{}`)) – Strings values define plotly.js patterns-shapes. Used to override `pattern_shape_sequences` to assign a specific patterns-shapes to lines corresponding with specific values. Keys in `pattern_shape_map` should be values in the column denoted by `pattern_shape`. Alternatively, if the values of `pattern_shape` are valid patterns-shapes names, the string `'identity'` may be passed to cause them to be used directly.

- **marginal** (*str* (https://docs.python.org/3/library/stdtypes.html#str)) – One of `'rug'`, `'box'`, `'violin'`, or `'histogram'`. If set, a subplot is drawn alongside the main plot, visualizing the distribution.
- **opacity** (*float* (https://docs.python.org/3/library/functions.html#float)) – Value between 0 and 1. Sets the opacity for markers.
- **orientation** (str, one of `'h'` for horizontal or `'v'` for vertical.) – (default `'v'` if `x` and `y` are provided and both continous or both categorical, otherwise `'v'`(‘h’) if `x`(`y`) is categorical and `y`(`x`) is continuous, otherwise `'v'`(‘h’) if only `x`(`y`) is provided)
- **barmode** (str (default `'relative'`)) – One of `'group'`, `'overlay'` or `'relative'` In `'relative'` mode, bars are stacked above zero for positive values and below zero for negative values. In `'overlay'` mode, bars are drawn on top of one another. In `'group'` mode, bars are placed beside each other.
- **barnorm** (str (default `None`)) – One of `'fraction'` or `'percent'`. If `'fraction'`, the value of each bar is divided by the sum of all values at that location coordinate. `'percent'` is the same but multiplied by 100 to show percentages. `None` will stack up all values at each location coordinate.
- **histnorm** (str (default `None`)) – One of `'percent'`, `'probability'`, `'density'`, or `'probability density'` If `None`, the output of `histfunc` is used as is. If `'probability'`, the output of `histfunc` for a given bin is divided by the sum of the output of `histfunc` for all bins. If `'percent'`, the output of `histfunc` for a given bin is divided by the sum of the output of `histfunc` for all bins and multiplied by 100. If `'density'`, the output of `histfunc` for a given bin is divided by the size of the bin. If `'probability density'`, the output of `histfunc` for a given bin is normalized such that it corresponds to the probability that a random event whose distribution is described by the output of `histfunc` will fall into that bin.
- **log_x** (boolean (default `False`)) – If `True`, the x-axis is log-scaled in cartesian coordinates.
- **log_y** (boolean (default `False`)) – If `True`, the y-axis is log-scaled in cartesian coordinates.
- **range_x** (*list of two numbers*) – If provided, overrides auto-scaling on the x-axis in cartesian coordinates.
- **range_y** (*list of two numbers*) – If provided, overrides auto-scaling on the y-axis in cartesian coordinates.
- **histfunc** (str (default `'count'` if no arguments are provided, else `'sum'`)) – One of `'count'`, `'sum'`, `'avg'`, `'min'`, or `'max'`. Function used to aggregate values for summarization (note: can be normalized with `histnorm`). The arguments to this function are the values of `y` (`x`) if `orientation` is `'v'` (`'h'`).
- **cumulative** (boolean (default `False`)) – If `True`, histogram values are cumulative.

- **nbins** (*int* (https://docs.python.org/3/library/functions.html#int)) – Positive integer. Sets the number of bins.
- **text_auto** (bool or string (default `False`)) – If `True` or a string, the x or y or z values will be displayed as text, depending on the orientation A string like `'.2f'` will be interpreted as a `texttemplate` numeric formatting directive.
- **title** (*str* (https://docs.python.org/3/library/stdtypes.html#str)) – The figure title.
- **template** (*str* (https://docs.python.org/3/library/stdtypes.html#str) *or dict* (https://docs.python.org/3/library/stdtypes.html#dict) *or plotly.graph_objects.layout.Template instance*) – The figure template name (must be a key in plotly.io.templates) or definition.
- **width** (int (default `None`)) – The figure width in pixels.
- **height** (int (default `None`)) – The figure height in pixels.

Returns:

Return type: plotly.graph_objects.Figure (plotly.graph_objects.html#plotly.graph_objects.Figure)