pandas.Series

class `pandas.Series`(*data=None*, *index=None*, *dtype=None*, *name=None*, *copy=False*, *fastpath=False*) [source]

One-dimensional ndarray with axis labels (including time series).

Labels need not be unique but must be a hashable type. The object supports both integer- and label-based indexing and provides a host of methods for performing operations involving the index. Statistical methods from ndarray have been overridden to automatically exclude missing data (currently represented as NaN).

Operations between Series (+, -, /, , *) align values based on their associated index values– they need not be the same length. The result index will be the sorted union of the two indexes.

Parameters:

data : *array-like, dict, or scalar value*
Contains data stored in Series
Changed in version 0.23.0: If data is a dict, argument order is maintained for Python 3.6 and later.

index : *array-like or Index (1d)*
Values must be hashable and have the same length as *data*. Non-unique index values are allowed. Will default to RangeIndex (0, 1, 2, …, n) if not provided. If both a dict and index sequence are used, the index will override the keys found in the dict.

dtype : *numpy.dtype or None*
If None, dtype will be inferred

copy : *boolean, default False*
Copy input data

Attributes

`T`	return the transpose, which is by definition self
`asobject`	Return object Series which contains boxed values.
`at`	Access a single value for a row/column label pair.
`axes`	Return a list of the row axis labels
`base`	return the base object if the memory of the underlying data is shared
`blocks`	(DEPRECATED) Internal property, property synonym for as_blocks()
`data`	return the data pointer of the underlying data
`dtype`	return the dtype object of the underlying data
`dtypes`	return the dtype object of the underlying data
`flags`	
`ftype`	return if the data is sparse|dense
`ftypes`	return if the data is sparse|dense
`hasnans`	return if I have any nans; enables various perf speedups
`iat`	Access a single value for a row/column pair by integer position.
`iloc`	Purely integer-location based indexing for selection by position.
`index`	The index (axis labels) of the Series.
`is_monotonic`	Return boolean if values in the object are monotonic_increasing
`is_monotonic_decreasing`	Return boolean if values in the object are monotonic_decreasing
`is_monotonic_increasing`	Return boolean if values in the object are monotonic_increasing
`is_unique`	Return boolean if values in the object are unique
`itemsize`	return the size of the dtype of the item of the underlying data
`ix`	A primarily label-location based indexer, with integer position fallback.
`loc`	Access a group of rows and columns by label(s) or a boolean array.
`nbytes`	return the number of bytes in the underlying data
`ndim`	return the number of dimensions of the underlying data, by definition 1
`shape`	return a tuple of the shape of the underlying data

`size`	return the number of elements in the underlying data
`strides`	return the strides of the underlying data
`values`	Return Series as ndarray or ndarray-like depending on the dtype

empty
imag
is_copy
name
real

Methods

`abs`()	Return a Series/DataFrame with absolute numeric value of each element.
`add`(other[, level, fill_value, axis])	Addition of series and other, element-wise (binary operator *add*).
`add_prefix`(prefix)	Prefix labels with string *prefix*.
`add_suffix`(suffix)	Suffix labels with string *suffix*.
`agg`(func[, axis])	Aggregate using one or more operations over the specified axis.
`aggregate`(func[, axis])	Aggregate using one or more operations over the specified axis.
`align`(other[, join, axis, level, copy, …])	Align two objects on their axes with the specified join method for each axis Index
`all`([axis, bool_only, skipna, level])	Return whether all elements are True, potentially over an axis.
`any`([axis, bool_only, skipna, level])	Return whether any element is True over requested axis.
`append`(to_append[, ignore_index, …])	Concatenate two or more Series.
`apply`(func[, convert_dtype, args])	Invoke function on values of Series.
`argmax`([axis, skipna])	(DEPRECATED) .. deprecated:: 0.21.0
`argmin`([axis, skipna])	(DEPRECATED) .. deprecated:: 0.21.0
`argsort`([axis, kind, order])	Overrides ndarray.argsort.
`as_blocks`([copy])	(DEPRECATED) Convert the frame to a dict of dtype -> Constructor Types that each has a homogeneous dtype.
`as_matrix`([columns])	(DEPRECATED) Convert the frame to its Numpy-array representation.
`asfreq`(freq[, method, how, normalize, …])	Convert TimeSeries to specified frequency.
`asof`(where[, subset])	The last row without any NaN is taken (or the last row without NaN considering only the subset of columns in the case of a DataFrame)
`astype`(dtype[, copy, errors])	Cast a pandas object to a specified dtype `dtype`.
`at_time`(time[, asof])	Select values at particular time of day (e.g.
`autocorr`([lag])	Lag-N autocorrelation
`between`(left, right[, inclusive])	Return boolean Series equivalent to left <= series <= right.
`between_time`(start_time, end_time[, …])	Select values between particular times of the day (e.g., 9:00-9:30 AM).
`bfill`([axis, inplace, limit, downcast])	Synonym for `DataFrame.fillna(method='bfill')`
`bool`()	Return the bool of a single element PandasObject.
`cat`	alias of `pandas.core.arrays.categorical.CategoricalAccessor`
`clip`([lower, upper, axis, inplace])	Trim values at input threshold(s).
`clip_lower`(threshold[, axis, inplace])	Return copy of the input with values below a threshold truncated.
`clip_upper`(threshold[, axis, inplace])	Return copy of input with values above given value(s) truncated.
`combine`(other, func[, fill_value])	Perform elementwise binary operation on two Series using given function with optional fill value when an index is missing from one Series or the other
`combine_first`(other)	Combine Series values, choosing the calling Series's values first.
`compound`([axis, skipna, level])	Return the compound percentage of the values for the requested axis
`compress`(condition, *args, **kwargs)	Return selected slices of an array along given axis as a

	Series
`consolidate`([inplace])	(DEPRECATED) Compute NDFrame with “consolidated” internals (data of each dtype grouped together in a single ndarray).
`convert_objects`([convert_dates, …])	(DEPRECATED) Attempt to infer better dtype for object columns.
`copy`([deep])	Make a copy of this object’s indices and data.
`corr`(other[, method, min_periods])	Compute correlation with *other* Series, excluding missing values
`count`([level])	Return number of non-NA/null observations in the Series
`cov`(other[, min_periods])	Compute covariance with Series, excluding missing values
`cummax`([axis, skipna])	Return cumulative maximum over a DataFrame or Series axis.
`cummin`([axis, skipna])	Return cumulative minimum over a DataFrame or Series axis.
`cumprod`([axis, skipna])	Return cumulative product over a DataFrame or Series axis.
`cumsum`([axis, skipna])	Return cumulative sum over a DataFrame or Series axis.
`describe`([percentiles, include, exclude])	Generates descriptive statistics that summarize the central tendency, dispersion and shape of a dataset’s distribution, excluding `NaN` values.
`diff`([periods])	First discrete difference of element.
`div`(other[, level, fill_value, axis])	Floating division of series and other, element-wise (binary operator *truediv*).
`divide`(other[, level, fill_value, axis])	Floating division of series and other, element-wise (binary operator *truediv*).
`divmod`(other[, level, fill_value, axis])	Integer division and modulo of series and other, element-wise (binary operator *divmod*).
`dot`(other)	Matrix multiplication with DataFrame or inner-product with Series objects.
`drop`([labels, axis, index, columns, level, …])	Return Series with specified index labels removed.
`drop_duplicates`([keep, inplace])	Return Series with duplicate values removed.
`dropna`([axis, inplace])	Return a new Series with missing values removed.
`dt`	alias of `pandas.core.indexes.accessors.CombinedDatetimelikeProperties`
`duplicated`([keep])	Indicate duplicate Series values.
`eq`(other[, level, fill_value, axis])	Equal to of series and other, element-wise (binary operator *eq*).
`equals`(other)	Determines if two NDFrame objects contain the same elements.
`ewm`([com, span, halflife, alpha, …])	Provides exponential weighted functions
`expanding`([min_periods, center, axis])	Provides expanding transformations.
`factorize`([sort, na_sentinel])	Encode the object as an enumerated type or categorical variable.
`ffill`([axis, inplace, limit, downcast])	Synonym for `DataFrame.fillna(method='ffill')`
`fillna`([value, method, axis, inplace, …])	Fill NA/NaN values using the specified method
`filter`([items, like, regex, axis])	Subset rows or columns of dataframe according to labels in the specified index.
`first`(offset)	Convenience method for subsetting initial periods of time series data based on a date offset.
`first_valid_index`()	Return index for first non-NA/null value.
`floordiv`(other[, level, fill_value, axis])	Integer division of series and other, element-wise (binary operator *floordiv*).
`from_array`(arr[, index, name, dtype, copy, …])	Construct Series from array.
`from_csv`(path[, sep, parse_dates, header, …])	(DEPRECATED) Read CSV file.
`ge`(other[, level, fill_value, axis])	Greater than or equal to of series and other, element-wise (binary operator *ge*).
`get`(key[, default])	Get item from object for given key (DataFrame column, Panel slice, etc.).
`get_dtype_counts`()	Return counts of unique dtypes in this object.
`get_ftype_counts`()	(DEPRECATED) Return counts of unique ftypes in this object.
`get_value`(label[, takeable])	(DEPRECATED) Quickly retrieve single value at passed index label

`get_values`()	same as values (but handles sparseness conversions); is a view
`groupby`([by, axis, level, as_index, sort, …])	Group series using mapper (dict or key function, apply given function to group, return result as series) or by a series of columns.
`gt`(other[, level, fill_value, axis])	Greater than of series and other, element-wise (binary operator *gt*).
`head`([n])	Return the first *n* rows.
`hist`([by, ax, grid, xlabelsize, xrot, …])	Draw histogram of the input series using matplotlib
`idxmax`([axis, skipna])	Return the row label of the maximum value.
`idxmin`([axis, skipna])	Return the row label of the minimum value.
`infer_objects`()	Attempt to infer better dtypes for object columns.
`interpolate`([method, axis, limit, inplace, …])	Interpolate values according to different methods.
`isin`(values)	Check whether *values* are contained in Series.
`isna`()	Detect missing values.
`isnull`()	Detect missing values.
`item`()	return the first element of the underlying data as a python scalar
`items`()	Lazily iterate over (index, value) tuples
`iteritems`()	Lazily iterate over (index, value) tuples
`keys`()	Alias for index
`kurt`([axis, skipna, level, numeric_only])	Return unbiased kurtosis over requested axis using Fisher's definition of kurtosis (kurtosis of normal == 0.0).
`kurtosis`([axis, skipna, level, numeric_only])	Return unbiased kurtosis over requested axis using Fisher's definition of kurtosis (kurtosis of normal == 0.0).
`last`(offset)	Convenience method for subsetting final periods of time series data based on a date offset.
`last_valid_index`()	Return index for last non-NA/null value.
`le`(other[, level, fill_value, axis])	Less than or equal to of series and other, element-wise (binary operator *le*).
`lt`(other[, level, fill_value, axis])	Less than of series and other, element-wise (binary operator *lt*).
`mad`([axis, skipna, level])	Return the mean absolute deviation of the values for the requested axis
`map`(arg[, na_action])	Map values of Series using input correspondence (a dict, Series, or function).
`mask`(cond[, other, inplace, axis, level, …])	Return an object of same shape as self and whose corresponding entries are from self where *cond* is False and otherwise are from *other*.
`max`([axis, skipna, level, numeric_only])	This method returns the maximum of the values in the object.
`mean`([axis, skipna, level, numeric_only])	Return the mean of the values for the requested axis
`median`([axis, skipna, level, numeric_only])	Return the median of the values for the requested axis
`memory_usage`([index, deep])	Return the memory usage of the Series.
`min`([axis, skipna, level, numeric_only])	This method returns the minimum of the values in the object.
`mod`(other[, level, fill_value, axis])	Modulo of series and other, element-wise (binary operator *mod*).
`mode`()	Return the mode(s) of the dataset.
`mul`(other[, level, fill_value, axis])	Multiplication of series and other, element-wise (binary operator *mul*).
`multiply`(other[, level, fill_value, axis])	Multiplication of series and other, element-wise (binary operator *mul*).
`ne`(other[, level, fill_value, axis])	Not equal to of series and other, element-wise (binary operator *ne*).
`nlargest`([n, keep])	Return the largest *n* elements.
`nonzero`()	Return the *integer* indices of the elements that are non-zero
`notna`()	Detect existing (non-missing) values.
`notnull`()	Detect existing (non-missing) values.
`nsmallest`([n, keep])	Return the smallest *n* elements.
`nunique`([dropna])	Return number of unique elements in the object.
`pct_change`([periods, fill_method, limit, freq])	Percentage change between the current and a prior element.
`pipe`(func, *args, **kwargs)	Apply func(self, *args, **kwargs)

`plot`	alias of `pandas.plotting._core.SeriesPlotMethods`
`pop`(item)	Return item and drop from frame.
`pow`(other[, level, fill_value, axis])	Exponential power of series and other, element-wise (binary operator *pow*).
`prod`([axis, skipna, level, numeric_only, …])	Return the product of the values for the requested axis
`product`([axis, skipna, level, numeric_only, …])	Return the product of the values for the requested axis
`ptp`([axis, skipna, level, numeric_only])	Returns the difference between the maximum value and the
`put`(*args, **kwargs)	Applies the *put* method to its *values* attribute if it has one.
`quantile`([q, interpolation])	Return value at the given quantile, a la numpy.percentile.
`radd`(other[, level, fill_value, axis])	Addition of series and other, element-wise (binary operator *radd*).
`rank`([axis, method, numeric_only, …])	Compute numerical data ranks (1 through n) along axis.
`ravel`([order])	Return the flattened underlying data as an ndarray
`rdiv`(other[, level, fill_value, axis])	Floating division of series and other, element-wise (binary operator *rtruediv*).
`reindex`([index])	Conform Series to new index with optional filling logic, placing NA/NaN in locations having no value in the previous index.
`reindex_axis`(labels[, axis])	(DEPRECATED) Conform Series to new index with optional filling logic.
`reindex_like`(other[, method, copy, limit, …])	Return an object with matching indices to myself.
`rename`([index])	Alter Series index labels or name
`rename_axis`(mapper[, axis, copy, inplace])	Alter the name of the index or columns.
`reorder_levels`(order)	Rearrange index levels using input order.
`repeat`(repeats, *args, **kwargs)	Repeat elements of an Series.
`replace`([to_replace, value, inplace, limit, …])	Replace values given in *to_replace* with *value*.
`resample`(rule[, how, axis, fill_method, …])	Convenience method for frequency conversion and resampling of time series.
`reset_index`([level, drop, name, inplace])	Generate a new DataFrame or Series with the index reset.
`rfloordiv`(other[, level, fill_value, axis])	Integer division of series and other, element-wise (binary operator *rfloordiv*).
`rmod`(other[, level, fill_value, axis])	Modulo of series and other, element-wise (binary operator *rmod*).
`rmul`(other[, level, fill_value, axis])	Multiplication of series and other, element-wise (binary operator *rmul*).
`rolling`(window[, min_periods, center, …])	Provides rolling window calculations.
`round`([decimals])	Round each value in a Series to the given number of decimals.
`rpow`(other[, level, fill_value, axis])	Exponential power of series and other, element-wise (binary operator *rpow*).
`rsub`(other[, level, fill_value, axis])	Subtraction of series and other, element-wise (binary operator *rsub*).
`rtruediv`(other[, level, fill_value, axis])	Floating division of series and other, element-wise (binary operator *rtruediv*).
`sample`([n, frac, replace, weights, …])	Return a random sample of items from an axis of object.
`searchsorted`(value[, side, sorter])	Find indices where elements should be inserted to maintain order.
`select`(crit[, axis])	(DEPRECATED) Return data corresponding to axis labels matching criteria
`sem`([axis, skipna, level, ddof, numeric_only])	Return unbiased standard error of the mean over requested axis.
`set_axis`(labels[, axis, inplace])	Assign desired index to given axis.
`set_value`(label, value[, takeable])	(DEPRECATED) Quickly set single value at passed label.
`shift`([periods, freq, axis])	Shift index by desired number of periods with an optional time freq
`skew`([axis, skipna, level, numeric_only])	Return unbiased skew over requested axis Normalized by N-1
`slice_shift`([periods, axis])	Equivalent to *shift* without copying data.
`sort_index`([axis, level, ascending, …])	Sort Series by index labels.
`sort_values`([axis, ascending, inplace, …])	Sort by the values.
`sortlevel`([level, ascending, sort_remaining])	(DEPRECATED) Sort Series with MultiIndex by chosen level.
`squeeze`([axis])	Squeeze length 1 dimensions.

`std`([axis, skipna, level, ddof, numeric_only])	Return sample standard deviation over requested axis.
`str`	alias of `pandas.core.strings.StringMethods`
`sub`(other[, level, fill_value, axis])	Subtraction of series and other, element-wise (binary operator *sub*).
`subtract`(other[, level, fill_value, axis])	Subtraction of series and other, element-wise (binary operator *sub*).
`sum`([axis, skipna, level, numeric_only, …])	Return the sum of the values for the requested axis
`swapaxes`(axis1, axis2[, copy])	Interchange axes and swap values axes appropriately
`swaplevel`([i, j, copy])	Swap levels i and j in a MultiIndex
`tail`([n])	Return the last *n* rows.
`take`(indices[, axis, convert, is_copy])	Return the elements in the given *positional* indices along an axis.
`to_clipboard`([excel, sep])	Copy object to the system clipboard.
`to_csv`([path, index, sep, na_rep, …])	Write Series to a comma-separated values (csv) file
`to_dense`()	Return dense representation of NDFrame (as opposed to sparse)
`to_dict`([into])	Convert Series to {label -> value} dict or dict-like object.
`to_excel`(excel_writer[, sheet_name, na_rep, …])	Write Series to an excel sheet
`to_frame`([name])	Convert Series to DataFrame
`to_hdf`(path_or_buf, key, **kwargs)	Write the contained data to an HDF5 file using HDFStore.
`to_json`([path_or_buf, orient, date_format, …])	Convert the object to a JSON string.
`to_latex`([buf, columns, col_space, header, …])	Render an object to a tabular environment table.
`to_msgpack`([path_or_buf, encoding])	msgpack (serialize) object to input file path
`to_period`([freq, copy])	Convert Series from DatetimeIndex to PeriodIndex with desired frequency (inferred from index if not passed)
`to_pickle`(path[, compression, protocol])	Pickle (serialize) object to file.
`to_sparse`([kind, fill_value])	Convert Series to SparseSeries
`to_sql`(name, con[, schema, if_exists, …])	Write records stored in a DataFrame to a SQL database.
`to_string`([buf, na_rep, float_format, …])	Render a string representation of the Series
`to_timestamp`([freq, how, copy])	Cast to datetimeindex of timestamps, at *beginning* of period
`to_xarray`()	Return an xarray object from the pandas object.
`tolist`()	Return a list of the values.
`transform`(func, *args, **kwargs)	Call function producing a like-indexed NDFrame and return a NDFrame with the transformed values
`transpose`(*args, **kwargs)	return the transpose, which is by definition self
`truediv`(other[, level, fill_value, axis])	Floating division of series and other, element-wise (binary operator *truediv*).
`truncate`([before, after, axis, copy])	Truncate a Series or DataFrame before and after some index value.
`tshift`([periods, freq, axis])	Shift the time index, using the index's frequency if available.
`tz_convert`(tz[, axis, level, copy])	Convert tz-aware axis to target time zone.
`tz_localize`(tz[, axis, level, copy, ambiguous])	Localize tz-naive TimeSeries to target time zone.
`unique`()	Return unique values of Series object.
`unstack`([level, fill_value])	Unstack, a.k.a.
`update`(other)	Modify Series in place using non-NA values from passed Series.
`valid`([inplace])	(DEPRECATED) Return Series without null values.
`value_counts`([normalize, sort, ascending, …])	Returns object containing counts of unique values.
`var`([axis, skipna, level, ddof, numeric_only])	Return unbiased variance over requested axis.
`view`([dtype])	Create a new view of the Series.
`where`(cond[, other, inplace, axis, level, …])	Return an object of same shape as self and whose corresponding entries are from self where *cond* is True and otherwise are from *other*.
`xs`(key[, axis, level, drop_level])	Returns a cross-section (row(s) or column(s)) from the Series/DataFrame.