lgimapy

Release 0.0.1

Jason R. Becker

June 10, 2019

Table of Contents

CHAPTER 1

Documentation for the Code

class lgimapy.index.**Index** (*index_df, name=''*)

Class for indexes built form *IndexBuilder*

Parameters
- **index_df** (*pd.DataFrame*) – Index DataFrame from *IndexBuilder.build()*.
- **name** (*str, default=''*) – Optional name of index.

df

Full index DataFrame.

Type pd.DataFrame

cuisps

List of all unique cusips in Index.

Type List[str]

dates

Sorted list of all dates in index.

Type List[datetime object]

day(date): Return Index or DataFrame of single speciifed date.

clean_treasuries(): Clean treasuries to on the run T bonds only.

market_value_weight(): Market value weight a specified column
vs entire index market value.

clean_treasuries ( )

Clean treasury index for building model curve.

cusips

Memoized unique cusips in Index.

dates

Memoized unique sorted dates in Index.

day (*date, as_index=False*)

Memoized call to a dict of single day DataFrame with date as key.

Parameters
- **date** (*datetime object*) – Date of daily DataFrame to return.

- **as_index** (*bool, default=False*) – If true, return an Index for specified day instead of a DataFrame.

Returns **df or Index** – DataFrame or Index for specified date.

Return type pd.DataFrame or *Index*

day_persistent_constituents (*date*, *as_index=False*)

Creates Index or DataFrames containing only the intersection of constituents which existed in the index both the day before and the specified day.

Parameters

- **date** (*datetime object*) – Date of daily DataFrame to return.
- **as_index** (*bool, default=False*) – If true, return an Index for specified day instead of a DataFrame.

Returns **df or Index** – DataFrame or Index for specified date.

Return type pd.DataFrame or *Index*

find_rating_changes (*rating_agency*)

Find rating changes of persistent index.

Parameters **rating_agency** (*{'SP', 'Moody', 'Fitch'}.*) – Rating agency to find changes for.

Returns **change_df** – DataFrame of all rating changes, with no index, columns are ['date', 'cusip', 'change'].

Return type pd.DataFrame

get_cusip_history (*cusip*)

Get full history for specified cusip.

Parameters **cusip** (*str*) – Specified cusip.

Returns **hist_df** – DataFrame with datetime index and Index.df columns for specified cusip.

Return type pd.DataFrame

get_value_history (*col*)

Get history of any column for all cusips in Index.

Parameters **col** (*str*) – Column from *Index.df* to build history for (e.g., 'OAS').

Returns **hist_df** – DataFrame with datetime index, CUSIP columns, and price values.

Return type pd.DataFrame

market_value_weight (*col*, *df=None*)

Market value weight a specified column vs entire index market value.

Parameters

- **col** (*str*) – Column name to weight, (e.g., 'OAS').
- **df** (*pd.DataFrame, default=None*) – DataFrame to use for analysis, use full Index.df if None.

Returns **mvw** – Market value weighting of specified column.

Return type float

subset_value_by_rating (*col*, *save=False*, *fid=None*, *path=None*)

Find the market vaue weighted value of specified column by rating.

Parameters
- **col** (*str*) – Column from Index.df to market value weight.
- **save** (*bool*) – If true, store resulting DataFrame to fid.
- **fid** (*str, default=None*) – File name to store file, by default is input *col*.
- **path** (*Path, default=None*) – Path to store file, by default is *./data/subset_by_rating*.

Returns **subset_df** – if save is false, return DataFrame of results with datetime index and numeric rating columns.

Return type pd.DataFrame

class lgimapy.index.**IndexBuilder**

Class to pull and clean index data from SQL database.

df

Full DataFrame of all cusips over loaded period.

Type pd.DataFrame

trade_dates

List of dates with bond data.

Type list[datetime]

loaded_dates

List of dates currently loaded by builder.

Type list[datetime]

build (*name=''*, *start=None*, *end=None*, *rating='IG'*, *currency=None*, *cusip=None*, *issuer=None*, *ticker=None*, *sector=None*, *subsector=None*, *municipals=True*, *treasuries=False*, *maturity=(None, None)*, *price=(None, None)*, *country_of_domicile=None*, *country_of_risk=None*, *amount_outstanding=(300, None)*, *issue_years=(None, None)*, *collateral_type=None*, *OAD=(None, None)*, *OAS=(None, None)*, *OASD=(None, None)*, *liquidity_score=(None, None)*, *credit_stats_only=False*, *credit_returns_only=False*, *financial_flag=None*, *special_rules=None*)

Build index with customized rules from *IndexBuilder.df*.

Parameters

Returns *Index* with specified rules.

Return type *Index*

load (*start=None*, *end=None*, *cusips=None*, *clean=True*, *dev=False*, *ret_df=False*, *local=False*, *data=None*)

Load data from SQL server. If end is not specified data is scraped through previous day. If neither start nor end are given only the data from previous day is scraped. Optionally load from local compressed format for increased performance or feed a DataFrame directly.

Parameters
- **start** (*str, datetime, default=None*) – Starting date for scrape.
- **end** (*str, datetime, default=None*) – Ending date for scrape.
- **cusips** (*List[str], default=None*) – List of cusips to specify for the load, by default load all.
- **clean** (*bool, default=True*) – If true, apply standard cleaning rules to loaded DataFrame.
- **dev** (*bool, default=False*) – If True, use development SQL query to load

all fields. **DO NOT USE IN PRODUCTION CODE**

- **ret_df** (*bool, default=False*) – If True, return loaded DataFrame.
- **local** (*bool, default=False*) – Load index from local binary file.

Returns **df** – DataFrame for specified date's index data if *ret_df* is true.

Return type pd.DataFrame

nearest_date (*date*)
Return trade date nearest to input date.

Parameters **date** (*datetime object*) – Input date.

Returns **t_date** – Trade date nearest to input date.

Return type datetime object

trade_dates
Datetime index array of dates with credit data.

lgimapy.index.**spread_diff** (*df1*, *df2*)
Calculate spread difference of index values from df1 to df2.

Parameters
- **df1** (*pd.DataFrame*) – Older index DataFrame.
- **df2** (*pd.DataFrame*) – Newer index DataFrame.

Returns **df**

Return type pd.DataFrame

lgimapy.index.**standardize_cusips** (*df*)
Standardize CUSIPs, converting cusips which changed name to most recent cusip value for full history.

Parameters **df** (*pd.DataFrame*) – DataFrame of selected index cusips.

Returns **df** – DataFrame with all CUSIPs updated to current values.

Return type pd.DataFrame

class lgimapy.models.**TreasuryCurve**
Class for loading treasury yield curve and calculating yields for specified dates and maturities.

trade_dates

Type List of traded dates.

get_yield(date, t): Get yield for specified date and maturities.

get_KRD_total_returns (*original_date*, *new_date*, *t*)
Reprice hypothetical par bond used for KRDs with a new yield curve.

get_KRD_yields
Memoized yields for specified date and maturities of [0.5, 2, 5, 10, 20, 30] years.

Parameters **date** (*datetime object*) – Date of yield curve.

Returns **yields** – Yields for [0.5, 2, 5, 10, 20, 30] years.

Return type [1 x 6] Array[float]

get_KRDs_and_coupons
Memoized key rate durations for specified date and maturities of [0.5, 2, 5, 10, 20, 30] years.

Parameters **date** (*datetime object*) – Date of yield curve.

Returns **krds** – KRDs for [0.5, 2, 5, 10, 20, 30] years.

Return type [1 x 6] Array[float]

get_yield (*date*, *t*)
Vectorized implementation to get yield(s) for a given date and maturities.

Parameters
- **date** (*datetime object*) – Date of yield curve.
- **t** (*float, nd.array[float]*) – Maturity or maturities (yrs) to return yields for.

Returns **y** – Yields for specified maturities.

Return type float, ndarray[float]

class lgimapy.models.**TreasuryCurveBuilder** (*ix*)
Class for fitting treasury curve and visualizing curve.

Parameters **ix** (*Index*) – Index class form IndexBuilder containing treasuries for a single date.

fit(): Fit curve to input data.

plot(): Plot fitted treasury curve.

fit (*method='price'*, *n=25*, *n_drop=10*, *threshold=12*, *solver='SLSQP'*, *verbose=0*)
Solve zero coupon bond curve given all bonds by minimizing the total squared error between the theoretical yield curve and market observed yield to maturities.
Methodolgy and notation taken from - https://www.jstatsoft.org/article/view/v036i01/v36i01.pdf
Parameters

plot (*trange=(0.1, 30)*, *indv_bonds=False*, *strips=False*, *ax=None*, *figsize=(8, 6)*)
Plot yield curve.

Parameters
- **trange** (*Tuple(float, float), default=(0.1, 30)*) – Range (min, max) in years to show.
- **indv_bonds** (*bool, default=False*) – If true, show individual bond yields as data points.
- **ax** (*matplotlib axis, default=None*) – Matplotlib axis to plot figure, if None one is created.
- **figsize** (*list or tuple, default=(6, 6)*) – Figure size.

save ( )
Save beta values to *../data/treasury_curve_params.csv*.

class lgimapy.models.**TBond** (*s*)
Class for treasury bond math and manipulation given current state of the bond.

Parameters **series** (*pd.Series*) – Single bond row from *index_builder* DataFrame.

coupon_dates

Type All coupon dates for given treasury.

coupon_years

Type Time to all coupons in years.

calculate_price(rfr): Calculate price of bond with given risk free rate.

calculate_price (*rfr*)
Calculate theoreticla price of the bond with given risk free rate.

Parameters **rfr** (*float*) – Continuously compounded risk free rate used to discount cash flows.

Returns **price** – Theoretical price ($) of bond.

Return type float

cash_flows
Return np.array of cash flows to be acquired on *coupon_dates*.

coupon_dates
Return list[datetime object] of timestamps for all coupons. Note that all coupons are assumed to be on either the 15th or last day of the month, business days and holidays are ignored.

coupon_days
Return np.array of time in years for all coupons.

coupon_years
Return np.array of time in years for all coupons.

theoretical_ytm (*price*)
Calculate yield to maturity of the bond using true coupon values and dates and specified price.

Parameters **price** (*float*) – Price of the bond to use when calculating yield to maturity.

Returns **ytm** – Yield to maturity of the bond at specified price.

Return type float

ytm
Memoized yield to maturity of the bond using true coupon values, dates, and dirty price.

Parameters **price** (*float, default=None*) – Price of the bond to use when calculating yield to maturity. If no price is given, the market value dirty price is used.

Returns **ytm** – Yield to maturity of the bond at specified price.

Return type float

lgimapy.bloomberg.**get_bloomberg_subsector** (*cusips*)
Get bloomberg subsector for list of cusips. First attempts to use saved *bloomberg_subsectors.json* file, updating the file for any cusip which is unsuccesful.

Parameters **cusips** (*str, List[str]*) – Cusip(s) to search bloomberg for subsectors.

Returns **subsectors** – List of bloomberg subsectors matching input cusips.

Return type List[str]

lgimapy.bloomberg.**update_subsector_json**( )

Update bloomberg subsector json file.

- *Index*
- *Module Index*
- *Search Page*

l

S

T

U

Y