asset-ification Documentation

Release 0.0.1

Benjamin M. Gross

July 01, 2014

Contents

Contents:

CHAPTER 1

README.md

1.1 Introduction

When I first constructed this library, I created a simple R-Squqared maximization optimization and was done with it. However, it lent it self to certain challenges, for example:

1. The output was categorical instead of probabilistic
2. There was no inclusion of “multi-asset” funds

For those reasons, it felt important to go back to the drawing board. Some of my key concerns with this library were:

1. Speed: Pulling down price series data from APIs is slow and cumbersome, especially when there are hundres of computations to fit a single price series. For that reason, relied heavily on the `HDFStore` filetype to store and pull price data
2. Probabilistic Outcomes instead of categorical: After spending some time with some Machine Learning Books, I wanted to change the outcome that “some price series is asset class `<blank>`” into a coherent process.

So that’s really what I’m attempting to do with this library...

1.2 Installation

```
git clone git@github.com:benjaminmgross/asset-ification.git #if you ssh
cd asset_ificaiton
python setup.py install
```

1.3 Up and Running

The testing and asset class detection modules run on the basis that:

1. There exists a local `HDFStore` of data prices on which fast and numerous computations can be run
2. There is a `.csv` of `trained_assets.csv`, to which the algorithm can learn different asset classes (I’ve already provided one for you in `/dat/trained_assets.csv`, if you don’t want to make your own).

So let’s get things setup (assuming you want to leverage the tedious hours I spent classifying the first three-hundred-some-odd ETFs).

1. Install the package

2. setup your `HDFStore` as follows (again, assuming you want to just use what I've done):

   ```
   $ ipython
   Python 2.7.6 (default, Mar 22 2014, 22:59:56)
   Type "copyright", "credits" or "license" for more information.

   IPython 1.2.1 -- An enhanced Interactive Python.
   ?         -> Introduction and overview of IPython's features.
   %quickref -> Quick reference.
   help      -> Python's own help system.
   object?   -> Details about 'object', use 'object??' for extra details.

   In [1]: import asset_ification as ai
   In [2]: trained_data = pandas.Series.from_csv("../dat/trained_data.csv",
   ...:        header = 0)
   In [3]: ai.asset_ification.setup_trained_hdfstore(trained_data, store_path)
   ```

`store_path` is just the string variable of where you'd like to store the HDFStore file. And that's it, now you can find out the probablities that some rando ticker (Ticker: RNDO) is a given asset class, e.g.

```
In [4]: ai.find_nearest_neighbors(RNDO_adj_close, store_path, trained_data)
```

1.4 To Do

-

CHAPTER 2

asset-ification

CHAPTER 3

Indices and tables

- *genindex*
- *modindex*
- *search*