Python Exercises - Data Science packages

Numpy, Pandas, Scipy, Matplotlib

Exercise 2.1: generate random variables

Using Numpy, create a function randlist(n,a,b) which returns a list of size n with random integers between a and b.

Exercise 2.2: Pandas to read CSV files

Using Pandas, create a function filter_liquid_options(csv_file) which:

- reads the csv file and converts into a pandas dataframe
- Converts the column Bid/Ask/Volume to floats
- Create a Mid Column ((Bid+Ask)/2)
- Filter only the options with a non zero volume

Use the function on the file “Option_SP500.csv” and print the 5 first rows of the resulting dataframe.

Exercise 2.3: Estimate a correlation matrix from stock prices

Using Pandas and Numpy, create functions get_clean_prices(list_futures) and compute_correl_matrix(list_futures) which take a list of futures as arguments and:

get_clean_prices:

- Reads each csv file corresponding to the future and convert it into a pandas dataframe
- Stores each dataframe into a main python dictionary named “dict_futures”
- Cleans dates which are not common to all futures (we should only keep dates which exist for all futures in argument)
- Returns a dataframe with the list of common dates as index and a column with the spot price for each future

compute_correl_matrix:

- Cleans spot prices using the function get_clean_prices
- Computes arithmetic returns for each future
- Computes and returns the Correlation matrix between returns

Test the function on the list of futures:
list_futures=["CAC_FUTURE", "FTSE100_FUTURE", "MSCI_EUROPE_FUTURE"]

Exercise 2.4: Generate multivariate gaussian variables
Using Numpy and and the scatter function of matplotlib.pyplot, create a function generate_bivariate_normal(means, stds, corr, N) which takes as argument a list of two means, a list of two standard deviations, a correlation, and which:

- generate N instances of each gaussian variable
- Display the correlated instances with a scatter plot

Test the function calling: generate_bivariate_normal([0,0], [0.2, 0.1], 0.9, 1000)

Exercise 2.5: plot a density distribution
Using the numpy and matplotlib packages:

- Create a vector of size 1000 of gaussian variables with mean 0 and variance 1
- Create a function gaussian_distribution(mu, sigma) which returns the density of the gaussian distribution
- Plot an histogram of the vector of 1000 gaussian variables, split in 100 bins
- Plot on the same graph the real density of the gaussian distribution

Exercise 2.6: linear regression
Using matplotlib, scipy and numpy, create a function compute_linear_regression(list_futures) which takes 2 futures as arguments, which:

- Cleans the futures spot prices using the previous function get_clean_prices
- Compute the intercept and slope of the regression between the futures' returns
- Plot the raw data (futures' return) and result of the linear regression

Test the function on list_futures=["CAC_FUTURE", "FTSE100_FUTURE"]

Exercise 2.7: Use pandas to make pivot tables
Using pandas, create a function taking a csv file as argument, which:

- Reads the csv file and loads it as a pandas dataframe
- Makes a pivot table showing the volume of options traded by option maturity and option type (call or put)

Test the function on the file "Option_SP500.csv"

<u>Exercise 2.8: Loading and reading a json file</u>
Using the json package, create a function get_json_file_name(json_file) which:

- Reads a json file
- Convert it into a python dictionary (using json.load)
- Returns the name of the dataset

Test the function on the file "FEXDZ2014.json"

<u>Exercise 2.9: Scipy.optimize</u>

Using scipy, create a function find_minimum(g, x0) which finds the minimum of a function g. Test the function on the function 0.5 * np.exp(-x * (1-x))