Value at Risk Calculation using SPARK

Introduction

VaR definition: Loss with a given probability in a given period.

For instance, "The VaR of my portfolio has a 1-week 5% of 1.000.000€" means that the probability of losing more than 1M€ is 5%. In other words, one out of 20 days I will lose 1.000.000€ or more.

Methods for calculation:

- Parametric calculation
- Historical simulation
- Montecarlo

We will use the Montecarlo method, which naturally is the "best" of theese three methods.

Goal of this assignment: Calculate the VaR with a time horizon of one week (5 work days) with a probability of 5% for a portfolio of instruments contained in symbols.txt

Montecarlo Method for VaR

The method is based in the assumption that we can estimate the return of the portfolio as function of a set of market factors.

$$r_i = c_i + \sum_{j=1}^{m} w_{ij} * f_j$$

Where *r* = returns and *f* = factor

We will use four factors to model the returns of each instrument:

- S&P 500
- NASDAQ
- Oil price
- US Treasury Bonds yield

This is a simple linear model. There is a different model for each instrument in our portfolio. Sometimes the factors are "featurized" to include non-linear relationships in the linear model, for instance adding the square and the square root of the factors to the list of factors. It depends on your ability for financial modelling. In this exercise we will "featurize" the factors adding the square and the square root of each factor to the factor vector.

To calculate this model, we could use standard scikit-learn functions since the datasets we are dealing with at this point are small. To make things more interesting (and if you want to earn extra points) you can use spark ML functions. You can find documentation here:

- Complete Spark ML python api doc: https://spark.apache.org/docs/latest/api/python/pyspark.ml.html

- Spark ML Linear Regression: https://spark.apache.org/docs/latest/api/python/pyspark.ml.html#pyspark.ml.regression.LinearRegression

Once we have a model of the returns based on factors, we need a way to sample the returns of each instrument, which means sampling the possible values of factors. For the sake of simplicity, we will assume that the distribution of the 4 factors is a normal distribution, which is more or less correct.

We could sample each factor independently but, in that case, we will be ignoring the correlation among the four factors. We will use a multivariate normal distribution. To get sample values we just we can just use *numpy* like this:

```
f1, f2, f3, f4 = numpy.random.multivariate_normal(mean, cov)
```

With the sample values, we can calculate the returns of each instrument

$$r_{it} = c_i + \sum_{j=1}^{m} w_{ij} * f_{tj}$$

Where *t* is the trial.

We will get a different return value for each trial. The VaR is the minimum of the worst 5% of the total losses of all the trials. The program should try 10000 trials that should be parallelized using spark.

Data Sources

Instruments

URL for download one instrument prices:

https://www.alphavantage.co/query?function=TIME_SERIES_DAILY&symbol=YOURSYMBOL&outputsize=full&apikey=CAJH46WI0QYW2RGK&datatype=csv

Get your apikey at https://www.alphavantage.co/support/#api-key

List of symbols: symbols.txt

Factors

S&P 500:

https://www.alphavantage.co/query?function=TIME_SERIES_DAILY&symbol=^GSPC&outputsize=full&apikey=CAJH46WI0QYW2RGK

NASDAQ:

https://www.alphavantage.co/query?function=TIME_SERIES_DAILY&symbol=NDAQ&outputsize=full&apikey=CAJH46WI0QYW2RGK

Oil Prices

https://www.quandl.com/api/v3/datasets/OPEC/ORB/data.csv?start_date=2010-02-21&end_date=2019-02-21&api_key=LwmAyD-1JHAzhMKKSvDX

US Treasury Bonds

https://www.quandl.com/api/v3/datasets/USTREASURY/YIELD.csv?start_date=2010-02-01&end_date=2019-02-21&api_key=LwmAyD-1JHAzhMKKSvDX

(For the last two, get your key at https://www.quandl.com/account/profile)

How to get the data with a python program

```
import urllib.request as url

remoteFile =
url.urlopen('https://www.quandl.com/api/v3/datasets/OPEC/ORB/data.csv?start_date=2010-02-21&end_date=2019-02-21&api_key=LwmAyD-1JHAzhMKKSvDX')

html = remoteFile.read().decode('ascii').splitlines()

print(html)
```

Important notes

- We are not interested in absolute price values but in returns over a time period, in this case a week or five work days. The return for a given day is calculated as

  `(price(day) – price(day – 5)) / price(day – 5)`

- All downloaded data should have the same starting and ending days. For this exercise get the data from 2008-01-01 to 2019-01-31. Make sure that all data have the same dates.

- The data can contain with empty values. Fill the empty dates with some suitable value using for instance nearby values

What to deliver

- The names of the team members

- The programs used to complete the exercise. The programs should be self-documented. The quality of the documentation is an important factor in the evaluation.

- A paper explaining your approach. It must contain:

    - The general approach
    - The method used to align the data coming from different sources (remember that all data must be available for the same dates) and the method used to fill the empty values.
    - We have assumed that the returns of factors follow a normal distribution. Show graphically that this is the case.
    - We have assumed that a linear model with “factorized” factors can represent all the instruments. Show how good is this assumption and discuss how can it be improved
    - The approach to parallelize with Spark