## Dataset Overview

You are given a dataset derived from financial time series. The time column gives you the date and time the data was observed.

Some things to note about the dataset –

1. The feature f0 is a return
   1. Let T1 and T2 be the times of 2 consecutive rows in the dataset
   2. Price1 and price2 and the prices of an asset observed at T1 and T2 respectively
   3. Then f0 at T2 = (price2-price1)/price1
   4. The asset time series is given to you in f24
2. There are some missing values in the dataset. Your model should be able to handle missing values
3. Occasionally, there could be outliers in these series. Your model should be able to handle such outliers

## Target Variable

The aim of the exercise is to predict the return of the asset f24 over the next 24 business hours. Where we don’t have a row at exactly the 24 business hours point, we will use the row immediately before the 24 business hour cutoff. Naturally, where we have the exact row 24 business hours afterwards, we just use it.

2 examples will highlight the definition -

1. Example 1 - the 11th row in the data pertains to 18-Nov-15 10:15.
   1. 24 business hours afterwards = 19-Nov-15 10:15
   2. But we don’t have the row with that exact time. So we take the row that falls immediately before our 24 hour cutoff i.e. 19-Nov-15 10:14 (row 89)
   3. Method 1 - We take f0 from row 12 to row 89, both inclusive, to create the target variable in row 11.
      1. Let us say a Pandas series s consists of N returns. To compute the cumulative return, we do (1+s).cumprod()-1.
      2. In other words if the n returns are r1, r2, r3…rn, then the cumulative return is (1+r1)\*(1+r2)\*(1+r3)…(1+rn)-1
      3. We apply the above formula to the series of f0 from row 12 to row 89 to get the target variable of row 11
      4. Please note we are not using f0 from row 11 to create the target in row 11 as we are interested in future returns and the f0 in row 11 is the return in the past – from 18-Nov-15 10:10 to 18-Nov-15 10:15
   4. Method 2 - Another way to create the target variable in row 11 would be to use f24 and apply the formula -
      1. Target in row 11 = (f24 in row 89 – f24 in row 11) / f24 in row 11
      2. In this case we are using the price of the asset (f24) in row 11 to create the target for row 11
   5. Both methods of computing the target should give you identical results
2. Example 2 – handling holidays
   1. Row 230 has a time of 20-Nov-15 15:28 (this is a Friday)
   2. The corresponding row 24 business hours later is on Monday, 23-Nov-15 15:28
   3. The closest row before that cutoff is row 308 with a time of 23-Nov-15 15:27
   4. Note: if Monday was a holiday (i.e. missing from the dataset), then we would have skipped over to Tuesday

## Evaluation Metric

The idea is to build a model that generalizes well on unseen data. We have embargoed some data that falls after the data that you have been given. We will be using this “hidden” data to evaluate the model “out of sample”. The evaluation metric is Root Mean Square Error (RMSE).

If y is the time series of the target and ypred is your prediction, then

RMSE = sqrt(mean( (y\_i – ypred\_i)^2 ))

An example is provided below –

y ypred

0.01 0.005

0.02 0.0225

0.01 -0.01

-0.02 0.01

In this case, RMSE = 0.018243149

## Model Template

You are supplied with a model template in Python – the language you are expected to use to complete the test. The parts where you need to insert you code are highlighted with a “todo” tag. Broadly, the things to do are to figure out –

1. What sort of feature engineering you want to do?
2. What class of models are best suited for the task?
3. How to properly cross validate your models?

## Things to be careful about

1. Lookahead bias - The number one mistake candidates do in this test is to introduce a lookahead bias in their features code. **In real life, when making a prediction at T, you will only know information up to T. Using any information that is not known at T while making your prediction introduces a lookahead.** Sometimes just by looking at the code, it is not very obvious where the lookahead comes from. That is why we strongly recommend you to add testing code to ensure your feature generation doesn’t have this bug. We will run our own tests on your code too (we intentionally don’t disclose the tests we run).
2. Overfitting – just like a subtle lookahead, this can be very subtle too. If there are elements of your cross validation that are not robust, you could end up introducing overfitting and making the model less generalizable out of sample.
3. sklearn convention for the predict function - the predict function should follow the usual sklearn convention that (if ypred is the returned object) –
   1. ypred is an np.array
   2. ypred has same shape as the data in path\_to\_test\_csv.
   3. Nth row in ypred is the prediction for the Nth row in the input matrix i.e. the order of the rows is preserved

If you don’t have predictions for some rows, you can put nulls. But the shape should be same as the data provided.

## Deliverables

Kindly provide the following for us to evaluate your model –

1. Filled out model template
2. A document that briefly summarizes your modeling thought process
   1. In this document, please also provide answers to the questions asked in the feedback section below

## Feedback

We are interested in collecting feedback on the test to look for ways on how to improve it. Kindly help us by answering the below questions –

1. Were the instructions in the test clear? If not, kindly highlight improvements you would have liked to see
2. Was the model template easy to work with? Would you suggest any changes to it?
3. Was the amount of data supplied enough for the modeling task? Would having more data have helped in terms of granularity or timespan?
4. How much time did you spend on the test in total?
5. How much was spent in –
   1. Exploring the dataset
   2. Feature generation
   3. Model Fitting and Validation
6. Did you experience any challenges while attempting this test?
7. Did you learn something new while working on the test?
8. What sort of approaches did you try? What worked and what didn’t work?