Third Homework Assignment (AWS)

Due Date: **6/5/2022** at **11:59pm**

AWS Information

This homework assignment must be completed using Amazon Web Services. The dataset will be made available in Wolf **/home/public/trading**. You must move the data to an s3 bucket that you own using the AWS CLI which was installed on Wolf. **The S3 bucket that hosts this data must be private.** The data is very confidential and thus it must not leave this bucket. You are allowed to create a small subset for debugging and testing, **But you may NOT** move it your laptop/mac and perform these tasks. For debugging and testing on your own laptop/mac, you must create a fictitious dataset (if you choose to do so). The assignment must be done in spark.

Submission Guidelines:

Please follow carefully the instructions posted on the course's github page when submitting your solutions (https://github.com/MSIA/bigdatacourse/blob/master/README.md). Failure to follow the instructions will result in lost points.

Deliverables:

1. Your spark source code file: name the file "Exercise1.py/scala/java"
2. Output of your spark job: name the file "Exercise1.txt"
   The output file must include MAPE for each walk forward step. It must also include at the end the average MAPE score, the maximum one, and the minimum one.

Problem

Data: The csv file includes a descriptive header. The data is about trading 'profit' and features of a security (it is not a stock). There is a timestamp, bar number, profit, feature values from val12 to val78, and trade_id. The interpretation of these values is unknown. You should treat them as black-box features and include them in your analysis. All other fields are irrelevant for this homework assignment.

For a trade_id value, you can sort by bar_num (which should also sort the samples by the time stamp). This gives you profits in the correct sequential order, i.e., as they accumulate in time. A sequence is a term relating to all samples aka bars with the same trade id.

Description: The goal of the assignment is to predict the profit at the bar level. You will predict on increments of 10. That is, use profit from bars 1-10 and available features to predict 11-20. You then have access to bars 1-20 (including their profit values) to predict 21-30 (etc).

For example, you have the first 10 bars, and then for bar 11 you have: the time stamp and the feature values. You need to predict the profit for bar 11.

Next, you have the first 10 bars, the feature values and time stamps for bars 11 and 12 (note that you should not assume you have the profit for bars 11 and 12), and you need to predict the profit for bar 12. You then continue in this manner until bar 20.

At bar 21, assume that all profits are available for bars 1 to 20. Now you need to make predictions for bars 20-30 without knowing the profits for these bars, etc.

Regarding the evaluation process, the data spans years from 2008 to 2015. You should evaluate your model in the walk forward fashion. Train your model on the first 6 months of data. Then perform inference on the next one month (month 7). And then move forward for 6 months, training on month 6-12, and testing on month 13**. You can think of this as a structured way to define your train/test split** (clearly within train you can further divide to train/validate; this is completely up to you).

Example: train Jan 2008 to Jun 2008, inference on July 2008; train July 2008 to December 2008, inference on January 2009, etc.

You are free to use whatever classification model and also feature engineering is completely up to you. You can either create your own spark EC2 cluster (ill-advised) or use EMR.

By the way, the real-world problem was not about predicting profits, but when to sell/buy which is a much harder nut to crack.

Feature availability structure

The training set can be defined as follows:
you have features and profits for 1-10

For bar 11 you now have additional features from 11. Predict profit for 11 (given features and profit for 1-10; features for 11)

To predict the profit of 12: predict from features derived in 1-12, profits 1-10

To predict the profit of 13: predict from features derived in 1-13, profits 1-10

To predict the profit of 14: predict from features derived in 1-14, profits 1-10

…

To predict the profit of 20: predict from features derived in 1-20, profits 1-10

Then:

To predict the profit of 21: features 1-21, profits 1-20

Note that for predicting the profit of bar 10001, you *can* generate features from bars 1-10001, and profits from 1-10000, but there is no requirement to use all available bars to generate features.

- The intuitive rule is that if you're predicting any given bar at any given time, you only have access to information known prior to that time, to predict that bar
    - This is the definition for "features derived in 1-12" for predicting the profit of bar 12
    - *"features" refers to columns val12 to val78*
- The unintuitive rule is that if you're predicting the profit anywhere from bar 11-20, you only have access to the profits in the model up to bar 10
    - This is a restriction on feature availability **only if you are using prior profits as a feature in your model**

Evaluation Structure (using MAPE):

The evaluation structure simply defines a train/test split. You train on a set of six months, and evaluate the next month. There will be a MAPE value for the 7th month. This continues for months 6-12, and a MAPE value for the 13th month.

If the evaluation structure was a random train/test split, it's just as likely that a bar from July 2008 (in the 7th month) is in the test set, as a bar from January 2008 (in the 1st month).

Instead, we are artificially selecting the train/test splits.