Fernando Gomes Papi

- Q1
- **a**: –
- **b**: Os autores definem *price discovery* como o processo em que novas informações são incorporadas no preço dos ativos. A metodologia dos autores é confrontar a liderança na movimentação de preço dos mercados spot (à vista) e de derivativos (contratos futuros), tanto em exchanges reguladas como não-reguladas. O trabalho utilliza de duas métricas de price discovery: Generalized Information Share e Component Share.

  A principal conclusão deste estudo é que a grande maioria do movimento de price discovery vem dos instrumentos derivativos em exchanges não-reguladas. A segunda grande conclusão do estudo é que choques de preço nas exchanges não reguladas levam de 4 a 5 minutos para serem absorvidas pelos contratos derivativos nas exchanges reguladas.

  Os autores especulam que isso deve-se ao maior volume apresentado por exchanges não-reguladas, sendo estas muitas vezes o único ponto de acesso ao mercado de players menores, mineradores, que muitas vezes podem ser considerados muito mais bem informados que players maiores como hedge funds. A hipótese de manipulação de mercado feitas através destas exchanges não pode ser descartada também. Um ponto positivo apontado pelos autores é que a liderança de price discovery por contratos futuros sobre o mercado spot é uma caracterísitca de mercados amadurecidos.

- **c**: –
- Q2
- **a**: Os modelos mais comuns de previsão de volatilidade são baseados em algoritmos de autoregressão, como ARCH e GARCH. De acordo com Hansen e Lude ("Forecasting Volatility using High Frequency Data", anexado), o principal problema introduzido por amostragem de alta frequência é a introdução de ruídos na mensuração do valor de um ativo que leva a um desvio de seu preço eficiente. Esse ruído pode ser devido à efeitos de liquidez, spread entre bid/ask e mensurações errôneas. Além disso, ruídos de alta frequência tendem a ser amplificados quando são derivados.

  O trabalho de Ghysels e Sinko ("Volatility forecasting and microstructure noise", anexado) faz uma análise empírica de diferentes metodologias de previsão na presença de ruído microestrutural, e aponta o modelo que se utiliza de médias e sub-amostragem do trabalho de Zhang et al. (2005) ("A Tale of Two Time Scales: Determining Integrated Volatility With Noisy High-Frequency Data", anexado) como o mais adequado. A conclusão deste trabalho é enfática: "*One important message of the article is that any time one has an impulse to sample sparsely, one can always do better with a multigrid method, regardless of the model or the quantity being estimated.*". Ou seja, que mesmo que ao tomar uma frequência de amostragem menor aparente melhorar os resultados obtidos, informação nunca deveria ser jogada fora.

  Este trabalho usa uma metodologia de separar as observações em diferentes "grids", fazendo uma média de diferentes estimadores baseados nestes grids. De certa forma,

isso é análogo a uma abordagem de **bagging** (bootstrap aggregating) em aprendizado estatístico, que treina N modelos fracos de aprendizado em paralelo visando reduzir a variância em datasets ruidosos.

- **b**: O trabalho "Volatility and Return Forecasting with High-Frequency and GARCH Models: Evidence for the Brazilian Market", em anexo, conclui que os modelos HAR (Heterogeneous Autoregressive) e 2-Comp são passíveis de se utilizar em modelagem de alta-frequência. Baseado nisso e nas conclusões do item acima, minha estratégia inicial seria de um algoritmo de bagging destes modelos.

- **c**: –

- Q3

  - **a - d**: Ver Jupyter Notebook Q3
  - **e**: Utilizei o algoritmo SHAP para explicar a modelagem binária proposta. Esse algoritmo é capaz de explicar modelos classificatórios agnósticos, apontando os pesos relativos para as features que expliquem a decisão do modelo.

    Baseado nesse algoritmo, nenhuma das features apresentadas no instante t tem poder explanatório sobre o fechamento do ativo no instante t+1.

    Além disso, a biblioteca sklearn possui um método de seleção de features, podendo-se utilizar o método ANOVA como teste estatístico. Nenhuma das features disponíveis em t (open, close, volume, MMS_3 e MMS_21) possuem um p-value onde pode-se rejeitar a hipótese nula.

    Uma terceira verificação seria um "reality check". Principalmente com as médias móveis em mente, é possível raciocinar que a tendência passada não necessariamente reflete a tendência futura, principalmente no curtíssimo prazo (já que estamos olhando para instantes de tempo de ordem de minuto).
  - **f**: Supondo que os dados estivessem periodicamente disponíveis via API, eu utilizaria serviços da Amazon Web Services para a arquitetura. As features propostas não são particularmente exigentes computacionalmente. Portanto é possível pensar em um microserviço baseado no Amazon Lambda, que permite a paralelização de tarefas e tem um custo bastante reduzido.

    É possível montar uma rotina de execução contínua com o serviço Eventbridge, que funciona de maneira análoga a uma CRONTAB (linux). O Eventbridge pode ser configurado para executar continuamente (p.ex. a cada minuto) ou baseado em regras (p.ex "Todo Domingo e Quarta às 20:00 UTC"). O Eventbridge pode disparar trabalhos assincronamente no serviço Amazon Lambda. O Lambda, por sua vez, é capaz de executar paralelamente as rotinas definidas a partir de um ponto de execução inicial, definido pelo usuário. O trabalho do Lambda, nesse caso, seria o de coletar os dados via API, calcular as features e armazená-las em um banco de dados, via API. A Amazon também disponibiliza serviços de bancos de dados, sendo o Amazon RDS um banco de dados relacional SQL (semelhante ao MySQL ou PostgreSQL) e o Amazon Dynamodb um banco de dados não relacional do tipo chave:valor.

A limitação do Lambda se dá no tempo de execução, que é limitado, e na configuração de bibliotecas externas a serem utilizadas nas rotinas, havendo um limite de tamanho do arquivo que o Lambda utiliza.

Uma opção viável seria utilizar o serviço de servidores virtuais, EC2. No EC2 há mais flexibilidade para implementar um servidor que execute as rotinas necessárias, utilizando de uma CRON para sincronizar os trabalhos.

Para tarefas mais exigentes computacionalmente, é possível utilizar o AWS Batch, sendo possível inclusive criar tarefas que utilizam GPU para processamento mais veloz