ĐỀ CƯƠNG ÔN TẬP

PHÂN TÍCH DỮ LIỆU LỚN TRONG TÀI CHÍNH

Dành cho sinh viên đại học ngành công nghệ tài chính, tài chính – ngân hàng

Big Data Analytics in Finance – a bilingual study guide

Hạ tầng dữ liệu lớn · Quy trình phân tích · Dữ liệu luồng
Học máy · Đầu tư và tín dụng · Regtech · Quản trị rủi ro AI

Bài tập có lời giải · Tự luận có dàn ý · Trắc nghiệm có đáp án · Mã Python và SQL chạy thật

Song ngữ Việt – Anh · Vietnamese – English

Phiên bản 1.0 – Tháng 11 năm 2026
https://tailieuso.vin

MỤC LỤC

CONTENTS

Chương 1. Tổng quan về dữ liệu lớn trong tài chính – *Chapter 1. Big data in finance: an overview* 7

1.1. Dữ liệu lớn là gì: năm chữ V – *1.1. What big data is: the five Vs* 7

1.2. Đo dung lượng: từ byte đến petabyte – *1.2. Measuring size: from bytes to petabytes* 10

1.3. Ba dạng dữ liệu: có cấu trúc, bán cấu trúc, phi cấu trúc – *1.3. Three forms of data: structured, semi-structured, unstructured* 13

1.4. Các nguồn dữ liệu lớn trong tài chính – *1.4. Big data sources in finance* 16

1.5. Trí tuệ nhân tạo, học máy, học sâu và dữ liệu lớn – *1.5. Artificial intelligence, machine learning, deep learning and big data* 18

1.6. Vì sao AI cần dữ liệu lớn, và dữ liệu lớn cần AI – *1.6. Why AI needs big data, and big data needs AI* 22

1.7. Lợi ích và rủi ro tổng quan – *1.7. Benefits and risks at a glance* 23

1.8. Bốn loại phân tích – *1.8. Four kinds of analytics* 27

1.9. Việc làm và kỹ năng – *1.9. Jobs and skills* 29

1.10. Ứng dụng ở tổ chức tài chính: phiếu một trang cho Trúc Xanh Pay – *1.10. Application in a financial organisation: a one-page brief for Trúc Xanh Pay* 30

1.11. Thực hành bằng Python – *1.11. Practising with Python* 32

1.12. Bài tập vận dụng có lời giải – *Applied exercises with solutions* 37

1.13. Câu hỏi tự luận và dàn ý trả lời – *Essay questions and answer outlines* 39

1.14. Câu hỏi trắc nghiệm – *Multiple-choice questions* 41

Chương 2. Hạ tầng và công cụ xử lý dữ liệu lớn – *Chapter 2. Big data infrastructure and tools* 43

2.1. Mở rộng dọc, mở rộng ngang và cụm máy – *2.1. Scaling up, scaling out and clusters* 44

2.2. Hệ thống tệp phân tán HDFS – *2.2. The distributed file system HDFS* 46

2.3. MapReduce: chia việc, xáo trộn, gom kết quả – *2.3. MapReduce: split, shuffle, gather* 49

2.4. Phân vùng dữ liệu và lệch dữ liệu – *2.4. Data partitioning and skew* 53

2.5. Nối dữ liệu phân tán – *2.5. Distributed joins* 56

2.6. Apache Spark: tính trong bộ nhớ và đánh giá lười – *2.6. Apache Spark: in-memory computing and lazy evaluation* 58

2.7. NoSQL và định lý CAP – *2.7. NoSQL and the CAP theorem* 61

2.8. Định dạng lưu trữ, hồ dữ liệu, kho dữ liệu và lakehouse – *2.8. Storage formats, data lake, data warehouse and lakehouse* 63

2.9. Điện toán đám mây, thuê ngoài và rủi ro tập trung – *2.9. Cloud computing, outsourcing and concentration risk* 66

2.10. Chọn công cụ theo bài toán và ứng dụng ở Ngân hàng Trúc Xanh – *2.10. Choosing tools by problem and an application at Trúc Xanh Bank* 69

2.11. Thực hành bằng Python – *2.11. Practising with Python* 71

2.12. Bài tập vận dụng có lời giải – *Applied exercises with solutions* 73
2.13. Câu hỏi tự luận và dàn ý trả lời – *Essay questions and answer outlines* 75
2.14. Câu hỏi trắc nghiệm – *Multiple-choice questions* 77

Chương 3. Quy trình phân tích và chất lượng dữ liệu lớn – *Chapter 3. The big data analytics process and data quality* 79
3.1. Quy trình phân tích dữ liệu lớn từ câu hỏi đến giám sát – *3.1. The big data analytics process from question to monitoring* 79
3.2. Viết câu hỏi phân tích tốt – *3.2. Writing a good analytical question* 81
3.3. Thu thập dữ liệu: giao diện lập trình, tệp lô, luồng sự kiện và thu thập web – *3.3. Collecting data: APIs, batch files, event streams and web scraping* 83
3.4. ETL, ELT và đường ống dữ liệu – *3.4. ETL, ELT and data pipelines* 84
3.5. Quản lý dữ liệu: chất lượng, tính đại diện và tính phù hợp – *3.5. Data management: quality, representativeness and relevance* 87
3.6. Kiểm tra chất lượng dữ liệu lớn – *3.6. Checking the quality of big data* 90
3.7. Xử lý giá trị thiếu và ngoại lai – *3.7. Handling missing values and outliers* 94
3.8. Gán nhãn và cấu trúc hóa dữ liệu cho học máy – *3.8. Labelling and structuring data for machine learning* 96
3.9. Lấy mẫu: ngẫu nhiên, phân tầng và hồ chứa – *3.9. Sampling: simple random, stratified and reservoir* 99
3.10. Khám phá dữ liệu ở quy mô lớn và nội dung phân tích trong tài chính – *3.10. Exploring data at scale and the content of financial analytics* 103
3.11. Thực hành bằng Python: giám sát chất lượng dữ liệu theo ngày – *3.11. Python practice: monitoring daily data quality* 107
3.12. Bài tập vận dụng có lời giải – *Applied exercises with solutions* 110
3.13. Câu hỏi tự luận và dàn ý trả lời – *Essay questions and answer outlines* 112
3.14. Câu hỏi trắc nghiệm – *Multiple-choice questions* 114

Chương 4. Dữ liệu luồng, thời gian thực và giám sát giao dịch – *Chapter 4. Streaming data, real-time analytics and transaction monitoring* 116
4.1. Xử lý theo lô, theo luồng và gần thời gian thực – *4.1. Batch, stream and near-real-time processing* 116
4.2. Sự kiện, thời điểm phát sinh, đến muộn và thủy vân – *4.2. Events, event time, late arrivals and watermarks* 118
4.3. Cửa sổ trên luồng: cố định, trượt và theo phiên – *4.3. Windows on a stream: tumbling, sliding and session* 120
4.4. Hàng đợi thông điệp và Kafka – *4.4. Message queues and Kafka* 123
4.5. Chỉ số tính dần trên luồng – *4.5. Indicators computed incrementally on a stream* 125
4.6. Thuật toán xấp xỉ bộ nhớ nhỏ: Bloom, Count-Min, HyperLogLog – *4.6. Small-memory approximate algorithms: Bloom, Count-Min, HyperLogLog* 127
4.7. Dữ liệu thị trường tần suất cao: tick, nến OHLC và VWAP – *4.7. High-frequency market data: ticks, OHLC candles and VWAP* 134
4.8. Giám sát giao dịch và phát hiện gian lận bằng trí tuệ nhân tạo (AI) – *4.8. Transaction monitoring and fraud detection with artificial intelligence (AI)* 136

4.9. Đo hiệu quả cảnh báo, ứng dụng và lập luận ủng hộ – phản đối – *4.9. Measuring alert quality, applications and arguments for and against* 140
4.10. Thực hành bằng Python – *4.10. Practice with Python* 143
4.11. Bài tập vận dụng có lời giải – *Applied exercises with solutions* 146
4.12. Câu hỏi tự luận và dàn ý trả lời – *Essay questions and answer outlines* 148
4.13. Câu hỏi trắc nghiệm – *Multiple-choice questions* 150

Chương 5. Phương pháp học máy và độ vững của mô hình trên dữ liệu lớn – *Chapter 5. Machine learning methods and model robustness at scale* 152

5.1. Ôn nhanh: đặc trưng, nhãn, ba tập dữ liệu và quá khớp – *5.1. A quick refresher: features, labels, three datasets and overfitting* 152
5.2. Học máy khi dữ liệu lớn: gradient ngẫu nhiên theo lô nhỏ và học trực tuyến – *5.2. Machine learning on big data: mini-batch stochastic gradient and online learning* 156
5.3. Phân cụm khách hàng bằng k-means – *5.3. Clustering customers with k-means* 160
5.4. Cây quyết định, rừng ngẫu nhiên, tăng cường, mạng nơ-ron và học sâu – *5.4. Decision trees, random forests, boosting, neural networks and deep learning* 164
5.5. Huấn luyện, kiểm định và kiểm tra theo thông lệ quản lý – *5.5. Training, validation and testing in supervisory practice* 165
5.6. Trôi dữ liệu, trôi khái niệm và chỉ số ổn định quần thể PSI – *5.6. Data drift, concept drift and the population stability index PSI* 169
5.7. Quản lý rủi ro mô hình: vòng đời, thẩm định và hướng dẫn của cơ quan quản lý – *5.7. Model risk management: life cycle, validation and supervisory guidance* 173
5.8. Tương quan không phải nhân quả và học vô nghĩa – *5.8. Correlation is not causation, and meaningless learning* 175
5.9. Rủi ro đuôi, kiểm tra sức chịu đựng và con người giám sát – *5.9. Tail risk, stress testing and human oversight* 177
5.10. Thực hành bằng Python – *5.10. Practice with Python* 180
5.11. Bài tập vận dụng có lời giải – *Applied exercises with solutions* 183
5.12. Câu hỏi tự luận và dàn ý trả lời – *Essay questions and answer outlines* 185
5.13. Câu hỏi trắc nghiệm – *Multiple-choice questions* 187

Chương 6. Dữ liệu lớn trong đầu tư và giao dịch – *Chapter 6. Big data in investment and trading* 189

6.1. Trí tuệ nhân tạo và dữ liệu lớn trong quản lý tài sản – *6.1. AI and big data in asset management* 189
6.2. Dữ liệu thay thế và chỉ số cảm xúc – *6.2. Alternative data and sentiment indices* 192
6.3. Giao dịch thuật toán, giao dịch tần suất cao và bánh xe thuật toán – *6.3. Algorithmic trading, high-frequency trading and algo wheels* 195
6.4. Chia lệnh lớn: TWAP, VWAP và chi phí thực hiện – *6.4. Slicing large orders: TWAP, VWAP and execution cost* 198
6.5. Kiểm tra ngược đúng cách – *6.5. Backtesting done properly* 202
6.6. Hệ quả ngoài ý muốn: bầy đàn, rút thanh khoản và sập chớp nhoáng – *6.6. Unintended consequences: herding, liquidity withdrawal and flash crashes* 205
6.7. Thao túng bằng thuật toán và cấu kết ngầm – *6.7. Algorithmic manipulation and tacit collusion* 207

6.8. Đầu tư ESG dùng AI – *6.8. ESG investing with AI* 211
6.9. Ứng dụng ở Công ty Chứng khoán Trúc Xanh – *6.9. Application at Trúc Xanh Securities* 213
6.10. Thực hành bằng Python – *6.10. Practising with Python* 215
6.11. Bài tập vận dụng có lời giải – *Applied exercises with solutions* 219
6.12. Câu hỏi tự luận và dàn ý trả lời – *Essay questions and answer outlines* 221
6.13. Câu hỏi trắc nghiệm – *Multiple-choice questions* 222

Chương 7. Dữ liệu lớn trong tín dụng, ngân hàng và bảo hiểm – *Chapter 7. Big data in credit, banking and insurance* 225
7.1. Tín dụng và bài toán chấm điểm bằng dữ liệu – *7.1. Credit and the data-driven scoring problem* 225
7.2. Dữ liệu truyền thống và dữ liệu thay thế – *7.2. Traditional and alternative data* 227
7.3. Dựng thẻ điểm: chia nhóm, trọng số chứng cứ và giá trị thông tin – *7.3. Building a scorecard: grouping, weight of evidence and information value* 229
7.4. Đánh giá thẻ điểm: ma trận nhầm lẫn và các chỉ số tách nhóm – *7.4. Evaluating a scorecard: confusion matrix and separation measures* 235
7.5. Minh bạch và công bằng khi cho vay – *7.5. Transparency and fairness in lending* 240
7.6. Giảm thiên lệch và khung pháp lý Việt Nam – *7.6. Reducing bias and the Vietnamese legal frame* 243
7.7. BigTech, tập trung dữ liệu và cạnh tranh – *7.7. BigTech, data concentration and competition* 245
7.8. Ngân hàng: cá nhân hóa, dự báo rời bỏ và bán chéo – *7.8. Banking: personalisation, churn prediction and cross-selling* 247
7.9. Bảo hiểm: định phí theo hành vi và phát hiện trục lợi – *7.9. Insurance: behaviour-based pricing and fraud detection* 249
7.10. Thực hành bằng Python – *7.10. Practising with Python* 252
7.11. Bài tập vận dụng có lời giải – *Applied exercises with solutions* 255
7.12. Câu hỏi tự luận và dàn ý trả lời – *Essay questions and answer outlines* 257
7.13. Câu hỏi trắc nghiệm – *Multiple-choice questions* 258

Chương 8. Regtech, Suptech và phân tích dữ liệu blockchain – *Chapter 8. Regtech, suptech and blockchain data analytics* 261
8.1. Regtech và Suptech: hai phía của một bàn làm việc – *8.1. Regtech and suptech: two sides of one desk* 261
8.2. Năm ứng dụng thường gặp – *8.2. Five common applications* 263
8.3. Báo cáo giám sát dạng dữ liệu và kiểm tra hợp lệ tự động – *8.3. Data-form supervisory reports and automatic validation* 266
8.4. Chuỗi khối, sổ cái phân tán và hợp đồng thông minh: ôn nhanh – *8.4. Blockchain, distributed ledgers and smart contracts: a quick review* 269
8.5. AI trong sản phẩm tài chính trên chuỗi khối – *8.5. AI in blockchain-based financial products* 272
8.6. Hợp đồng thông minh, oracle và AI – *8.6. Smart contracts, oracles and AI* 274
8.7. Hợp đồng tự học, DeFi, DAO và câu hỏi ai chịu trách nhiệm – *8.7. Self-learning contracts, DeFi, DAOs and who is responsible* 277

8.8. Phân tích dữ liệu trên chuỗi: đồ thị ví với ví – *8.8. On-chain analysis: the wallet-to-wallet graph* 279
8.9. Đo hoạt động trên chuỗi: ví hoạt động, khối lượng, độ tập trung – *8.9. Measuring on-chain activity: active wallets, volume, concentration* 282
8.10. Thử nghiệm có kiểm soát và tình huống ở Trúc Xanh Pay – *8.10. Controlled testing and a case at Trúc Xanh Pay* 283
8.11. Thực hành bằng Python – *8.11. Practice with Python* 284
8.12. Bài tập vận dụng có lời giải – *Applied exercises with solutions* 289
8.13. Câu hỏi tự luận và dàn ý trả lời – *Essay questions and answer outlines* 291
8.14. Câu hỏi trắc nghiệm – *Multiple-choice questions* 293

Chương 9. Quản trị dữ liệu và AI: quyền riêng tư, giải thích được, trách nhiệm – *Chapter 9. Data and AI governance: privacy, explainability and accountability* 295
9.1. Quyền riêng tư và bảo mật dữ liệu khi phân tích dữ liệu lớn – *9.1. Privacy and confidentiality when analysing big data* 295
9.2. Ẩn danh hóa, tái nhận dạng và k-ẩn danh – *9.2. Anonymisation, re-identification and k-anonymity* 298
9.3. Quyền riêng tư vi sai, học liên kết và dữ liệu tổng hợp – *9.3. Differential privacy, federated learning and synthetic data* 302
9.4. Giải thích được của mô hình AI – *9.4. Explainability of AI models* 306
9.5. Kiểm toán được và công bố thông tin cho khách hàng – *9.5. Auditability and disclosure to customers* 310
9.6. Quản trị hệ thống AI và trách nhiệm giải trình – *9.6. Governing AI systems and accountability* 313
9.7. Khung pháp lý và hoạt động chính sách về AI trong tài chính – *9.7. The legal frame and policy activity on AI in finance* 316
9.8. Khuyến nghị chính sách – *9.8. Policy recommendations* 318
9.9. Pháp luật Việt Nam và bảng tự kiểm trước khi đưa mô hình vào dùng – *9.9. Vietnamese law and the self-check before a model goes live* 320
9.10. Thực hành bằng Python – *9.10. Practising with Python* 322
9.11. Bài tập vận dụng có lời giải – *Applied exercises with solutions* 326
9.12. Câu hỏi tự luận và dàn ý trả lời – *Essay questions and answer outlines* 328
9.13. Câu hỏi trắc nghiệm – *Multiple-choice questions* 330

Chương 10. Trình bày kết quả và tình huống phân tích dữ liệu lớn ở thị trường, tổ chức tài chính – *Chapter 10. Presenting results and big data case studies in financial markets and institutions* 332
10.1. Bắt đầu từ quyết định: trình bày kết quả cho người ra quyết định – *10.1. Start from the decision: presenting results to the person who decides* 332
10.2. Chọn biểu đồ theo câu hỏi – *10.2. Choosing a chart by the question* 334
10.3. Bảng điều khiển và kể chuyện bằng dữ liệu – *10.3. Dashboards and data storytelling* 336
10.4. Khoảng tin cậy và cỡ mẫu khi báo cáo một tỷ lệ – *10.4. Confidence intervals and sample size when reporting a rate* 339

10.5. Kiểm định A/B cho sản phẩm ngân hàng số – *10.5. A/B testing for digital banking products* 341
10.6. Tình huống ở thị trường tài chính – *10.6. Case studies in the financial market* 345
10.7. Tình huống ở tổ chức tài chính – *10.7. Case studies in financial institutions* 351
10.8. Dự án tổng hợp của Bảo Ngọc: từ câu hỏi của chị Thảo đến báo cáo một trang – *10.8. Bảo Ngọc's capstone project: from chị Thảo's question to a one-page report* 358
10.9. Thực hành bằng Python – *10.9. Practising with Python* 361
10.10. Bài tập vận dụng có lời giải – *Applied exercises with solutions* 364
10.11. Câu hỏi tự luận và dàn ý trả lời – *Essay questions and answer outlines* 365
10.12. Câu hỏi trắc nghiệm – *Multiple-choice questions* 367

Phụ lục A. Đáp án câu hỏi trắc nghiệm – *Appendix A. Answer key* 369

Phụ lục B. Công thức – *Appendix B. Formulas* 374

Phụ lục C. Thuật ngữ Việt – Anh – *Appendix C. Vietnamese–English glossary* 381

Phụ lục D. Tài liệu tham khảo – *Appendix D. References* 388

Chương 1. Tổng quan về dữ liệu lớn trong tài chính – *Chapter 1. Big data in finance: an overview*

Mục tiêu chương – *Chapter objectives*

- Mô tả dữ liệu lớn bằng năm chữ V (khối lượng, tốc độ, đa dạng, tính xác thực, giá trị) và cho mỗi chữ một ví dụ ở ngân hàng hoặc công ty chứng khoán.
  Describe big data with the five Vs (volume, velocity, variety, veracity, value) and give each one an example from a bank or a securities company.
- Ước dung lượng dữ liệu giao dịch một năm bằng phép nhân, đổi từ byte lên các đơn vị lớn hơn và tính cả số bản sao.
  Estimate a year of transaction data by multiplication, convert from bytes up to larger units, and count the copies.
- Phân biệt dữ liệu có cấu trúc, bán cấu trúc, phi cấu trúc và kể tám nhóm nguồn dữ liệu trong tài chính.
  Tell structured, semi-structured and unstructured data apart and name eight groups of data sources in finance.
- Xếp đúng trí tuệ nhân tạo (AI, artificial intelligence), học máy (ML, machine learning), học sâu và dữ liệu lớn; chọn loại phân tích và nêu lợi ích, rủi ro, kỹ năng.
  Place artificial intelligence (AI), machine learning (ML), deep learning and big data correctly; choose a kind of analytics and state benefits, risks and skills.
- Dùng Python chuẩn đọc tệp lớn theo từng dòng bằng generator, đếm, cộng dồn và đo tỷ lệ dòng có vấn đề.
  Use standard Python to read a large file line by line with a generator, count, accumulate and measure the share of problem rows.

1.1. Dữ liệu lớn là gì: năm chữ V – *1.1. What big data is: the five Vs*

Ngày đầu đi thực tập, Bảo Ngọc ngồi cạnh anh Quân, trưởng nhóm kỹ sư dữ liệu lớn của Ngân hàng Thương mại Cổ phần (TMCP) Trúc Xanh. Anh mở một màn hình cho bạn xem ví Trúc Xanh Pay ghi thêm bao nhiêu dòng trong mỗi phút. Con số nhảy liên tục. Bảo Ngọc hỏi: bao nhiêu dòng thì gọi là dữ liệu lớn? Anh Quân đáp rằng không có mốc cố định, mà phải xem dữ liệu đó có làm công cụ quen thuộc bó tay hay không.

On her first day as an intern, Bảo Ngọc sits beside Mr Quân, head of the big data engineering team at Trúc Xanh Joint Stock Commercial Bank. He opens a screen showing how many rows the Trúc Xanh Pay wallet adds every minute. The number keeps jumping. Bảo Ngọc asks how many rows make data big. Mr Quân says there is no fixed line; what matters is whether the data defeats the tools you already know.

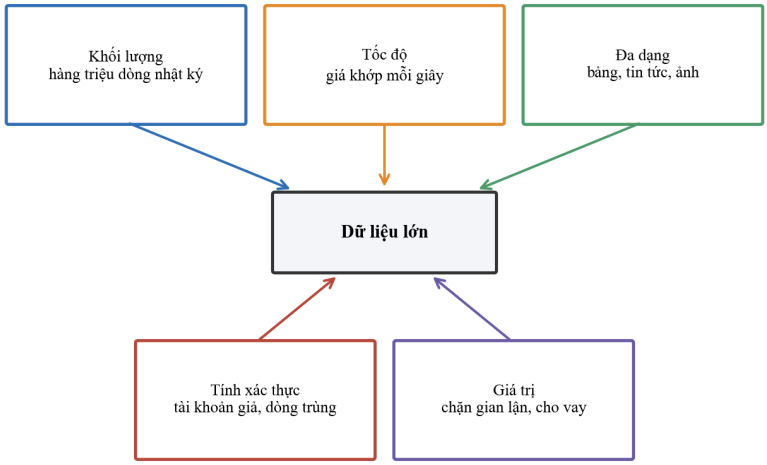
Dữ liệu lớn (big data) là dữ liệu quá nhiều, đến quá nhanh, quá lẫn lộn các dạng hoặc quá khó tin, nên một bảng tính hay một máy chủ cơ sở dữ liệu thông thường xử lý không nổi. Người ta mô tả nó bằng các chữ V. Laney (2001) nêu ba chữ đầu là khối lượng, tốc độ và đa dạng. Về sau người ta thêm tính xác thực và giá trị, thành năm chữ V.

Big data is data that is too plentiful, arrives too fast, mixes too many forms or is too hard to trust for an ordinary spreadsheet or database server to cope. People describe it with the letter V.

Laney (2001) named the first three: volume, velocity and variety. Later authors added veracity and value, which makes five Vs.

Báo cáo của Tổ chức Hợp tác và Phát triển Kinh tế (OECD, 2021) liệt kê bốn chữ V đầu và nhấn mạnh tính xác thực. Lý do là người dùng khó biết bộ dữ liệu đã đủ và đáng tin hay chưa, nên phải kiểm từng trường hợp. Hình 1.1 gom năm chữ V quanh khái niệm dữ liệu lớn.

The report of the Organisation for Economic Co-operation and Development (OECD, 2021) lists the first four Vs and stresses veracity. The reason is that users find it hard to tell whether a dataset is complete and reliable, so each case needs checking. Figure 1.1 gathers the five Vs around the idea of big data.



Hình 1.1. Hộp giữa là dữ liệu lớn. Mỗi mũi tên đi tới một chữ V, kèm một ví dụ ở ngân hàng. Nhìn hộp đỏ và hộp tím ở hàng dưới: hai chữ V này hay bị bỏ quên nhất.

Figure 1.1. *The middle box is big data. Each arrow reaches one V with a bank example. Look at the red and purple boxes in the bottom row: these two Vs are the ones most often forgotten.*

Bảng tiếp theo cho mỗi chữ V một lời giải thích bằng ngôn ngữ thường và một ví dụ ở ngân hàng, một ví dụ ở công ty chứng khoán. Tick là dòng dữ liệu ghi một lần khớp lệnh: giá khớp, số lượng và thời điểm. Mã KLP trong ví dụ là mã tự đặt.

The next table gives each V a plain-language meaning, one bank example and one securities example. A tick is one row of data for one matched trade: the price, the quantity and the time. The code KLP in the example is a made-up ticker.

Chữ V *The V*	**Nghĩa thường** *Plain meaning*	**Ví dụ ngân hàng** *Bank example*	**Ví dụ chứng khoán** *Securities example*
Khối lượng *Volume*	Nhiều đến mức một máy không chứa nổi *So much that one machine cannot hold it*	Nhật ký ứng dụng, 12 triệu sự kiện mỗi ngày *App logs, 12 million events a day*	Mọi lệnh đặt trong nhiều năm *Every order placed over many years*

Chữ V *The V*	**Nghĩa thường** *Plain meaning*	**Ví dụ ngân hàng** *Bank example*	**Ví dụ chứng khoán** *Securities example*
Tốc độ *Velocity*	Đến liên tục và phải xử lý kịp *Arrives nonstop and must be handled in time*	Chặn giao dịch thẻ nghi ngờ trong 200 mili giây *Block a suspect card payment within 200 milliseconds*	Mỗi lần khớp mã KLP sinh một tick *Each match of KLP creates a tick*
Đa dạng *Variety*	Nhiều dạng, nhiều nguồn *Many forms and sources*	Bảng giao dịch, email khiếu nại, ảnh giấy tờ *Transaction tables, complaint emails, document photos*	Giá, báo cáo tài chính, tin tức *Prices, financial statements, news*
Tính xác thực *Veracity*	Đáng tin đến đâu *How far it can be trusted*	Tài khoản mở bằng thông tin giả, dòng trùng *Accounts opened with false details, repeated rows*	Tin đồn trên mạng bị đọc như tin thật *A rumour online read as real news*
Giá trị *Value*	Dẫn tới quyết định có ích *Leads to a useful decision*	Giảm tổn thất do gian lận *Lower losses from fraud*	Phát hiện lệnh bất thường sớm hơn *Spot odd orders sooner*

Muốn biết một nguồn có phải dữ liệu lớn hay chưa, bạn làm năm bước sau. Bước đầu tiên là phép đếm thô; bốn bước còn lại cho biết nguồn đó thuộc chữ V nào.

To decide whether a source counts as big data, take five steps. The first is a rough count; the other four show which Vs the source touches.

1. Đếm lượng dữ liệu sinh ra mỗi ngày: số bản ghi nhân với số byte mỗi bản ghi.
   Count the data created per day: records times bytes per record.
2. Đo nhịp: chia số bản ghi cho số giây hoạt động, rồi hỏi hệ thống được phép trả lời trong bao lâu.
   Measure the pace: divide records by active seconds, then ask how soon the system must answer.
3. Liệt kê dạng dữ liệu: bảng, dòng JSON, văn bản, ảnh, giọng nói.
   List the forms of data: tables, JSON lines, text, images, speech.
4. Lấy một mẫu nhỏ, đếm dòng trùng, dòng thiếu, dòng vô lý để biết độ tin cậy.
   Take a small sample and count repeated, missing and absurd rows to gauge reliability.
5. Ghi ra quyết định mà nguồn này phục vụ. Không có quyết định nào thì nguồn chưa có giá trị.
   Write down the decision this source serves. With no decision, the source has no value yet.

Làm mẫu với nguồn lệnh của Công ty Chứng khoán Trúc Xanh. Bước 1: 900.000 lệnh mỗi ngày, mỗi lệnh 150 byte, được 135 triệu byte, tức 135 megabyte (MB) mỗi ngày. Bước 2: một phiên dài 4,5 giờ, tức 16.200 giây, nên trung bình 55,6 lệnh mỗi giây; lúc cao điểm gấp 5 lần là khoảng 278 lệnh mỗi giây. Bước 3: bảng lệnh có cấu trúc, kèm ghi

chú của môi giới là văn bản tự do. Bước 4: trong 1.000 lệnh lấy mẫu có 12 lệnh trùng mã, tỷ lệ 1,2%. Bước 5: nguồn này phục vụ việc phát hiện lệnh bất thường.

Worked example with the order source of Trúc Xanh Securities. Step 1: 900,000 orders a day at 150 bytes each gives 135 million bytes, or 135 megabytes (MB) a day. Step 2: a session lasts 4.5 hours, or 16,200 seconds, so the average is 55.6 orders per second; at a peak five times higher it is about 278 orders per second. Step 3: the order table is structured, and the broker's notes are free text. Step 4: in a sample of 1,000 orders, 12 repeat a reference, a rate of 1.2%. Step 5: the source serves the detection of odd orders.

Cách đọc kết quả: 135 MB mỗi ngày thì một máy vẫn chứa được. Nguồn lệnh này chỉ thành bài toán dữ liệu lớn khi nhìn qua chữ V tốc độ, vì 278 lệnh mỗi giây lúc cao điểm buộc hệ thống phải xử lý theo luồng.

Reading the result: 135 MB a day still fits on one machine. This order source becomes a big data problem only through the velocity V, because 278 orders a second at the peak force the system to process a stream.

> **LỖI HAY GẶP / COMMON MISTAKE –** Nghĩ dữ liệu lớn nghĩa là phải có hàng tỷ dòng. Một luồng nhỏ nhưng phải trả lời trong 200 mili giây vẫn là bài toán dữ liệu lớn vì chữ V tốc độ. Ngược lại, một tệp 50 gigabyte (GB) đọc một lần mỗi tháng thì chỉ cần kiên nhẫn.
> *Thinking big data means billions of rows. A small stream that must be answered in 200 milliseconds is still a big data problem because of velocity. A file of 50 gigabytes (GB) read once a month only needs patience.*

1.2. Đo dung lượng: từ byte đến petabyte – *1.2. Measuring size: from bytes to petabytes*

Đo dung lượng giống đo nước. Chai nước, can nước, bồn nước, hồ nước: mỗi bậc lớn hơn bậc trước 1.000 lần. Đơn vị gốc là byte, đủ để lưu một chữ cái hoặc một chữ số. Các đơn vị lớn hơn đều là bội của 1.000 byte.

Measuring size works like measuring water: a bottle, a jerrycan, a tank, a lake, each step 1,000 times the one before. The base unit is the byte, enough to store one letter or one digit. Every larger unit is a multiple of 1,000 bytes.

Đơn vị *Unit*	**Bằng** *Equals*	**Hình dung** *Picture it*	**Ở Trúc Xanh** *At Trúc Xanh*
Byte (B) *Byte (B)*	1 B	Một chữ cái *One letter*	Bản ghi ví dài 120 B *A wallet record of 120 B*
Kilobyte (KB) *Kilobyte (KB)*	1.000 B *1,000 B*	Một email ngắn *A short email*	Khoảng 8 bản ghi ví *About 8 wallet records*
Megabyte (MB) *Megabyte (MB)*	1.000 KB *1,000 KB*	Một ảnh chụp giấy tờ *A photo of a document*	Giao dịch ví trong vài giây *A few seconds of wallet trades*
Gigabyte (GB) *Gigabyte (GB)*	1.000 MB *1,000 MB*	Một bộ phim nén *A compressed film*	Khoảng 8,3 triệu bản ghi ví *About 8.3 million wallet records*

Đơn vị *Unit*	**Bằng** *Equals*	**Hình dung** *Picture it*	**Ở Trúc Xanh** *At Trúc Xanh*
Terabyte (TB) *Terabyte (TB)*	1.000 GB *1,000 GB*	Một ổ cứng cỡ lớn *A large hard drive*	5.556 ngày giao dịch ví *5,556 days of wallet trades*
Petabyte (PB) *Petabyte (PB)*	1.000 TB *1,000 TB*	Một trung tâm dữ liệu nhỏ *A small data centre*	Khoảng 707 năm của cả bốn nguồn *About 707 years of all four sources*

Có hai điều dễ nhầm. Thứ nhất, byte (chữ B hoa) khác bit (chữ b thường): một byte gồm 8 bit, tốc độ mạng thường tính bằng bit còn dung lượng tệp tính bằng byte. Thứ hai, một số hệ điều hành lấy 1 KB bằng 1.024 byte. Các phép tính trong sách này dùng 1 KB bằng 1.000 byte để dễ nhẩm; khi báo cáo ra ngoài bạn phải ghi rõ cách quy đổi đã dùng.

Two things trip people up. First, a byte (capital B) is not a bit (lower-case b): one byte holds 8 bits, network speed is usually quoted in bits and file size in bytes. Second, some operating systems take 1 KB as 1,024 bytes. The sums in this book use 1 KB as 1,000 bytes so they are easy to do in your head; when you report to others, say which conversion you used.

Ước dung lượng chỉ cần một phép nhân. Gọi N là số bản ghi mỗi ngày, b là số byte mỗi bản ghi, t là số ngày ghi trong năm. Dung lượng một năm D tính theo công thức sau.

Estimating storage takes one multiplication. Let N be the records per day, b the bytes per record and t the days recorded in a year. Annual storage D follows from the formula.

$$D = N * b * t \tag{1.1}$$

Làm tay với ví Trúc Xanh Pay. Mỗi giao dịch là một bản ghi 120 byte gồm mã giao dịch, hai mã ví, thời điểm, số tiền, loại, trạng thái và ghi chú ngắn. Ví ghi 1.500.000 giao dịch mỗi ngày, quanh năm.

Worked by hand for the Trúc Xanh Pay wallet. Each transaction is a 120-byte record holding a reference, two wallet codes, the time, the amount, the type, the status and a short note. The wallet records 1,500,000 transactions a day, all year.

1. Một ngày: 1.500.000 × 120 = 180.000.000 byte, tức 180 MB.
   One day: 1,500,000 × 120 = 180,000,000 bytes, which is 180 MB.
2. Một năm 365 ngày: 180 MB × 365 = 65.700 MB.
   One 365-day year: 180 MB × 365 = 65,700 MB.
3. Đổi đơn vị: 65.700 MB chia 1.000 được 65,7 GB.
   Convert: 65,700 MB divided by 1,000 gives 65.7 GB.
4. Hệ thống thường giữ 3 bản sao để chống mất dữ liệu. Nhân với 3 được 197,1 GB.
   Systems usually keep 3 copies against loss. Times 3 gives 197.1 GB.

Nghĩa là riêng giao dịch ví, một năm vừa một ổ cứng nhỏ. Bây giờ làm cho cả bốn nguồn của Trúc Xanh: giao dịch ví, nhật ký ứng dụng, lệnh chứng khoán và giá theo phút. Đoạn mã dưới đổi đơn vị tự động và hỏi thêm: nếu dữ liệu mỗi năm gấp 1,5 lần năm trước, bao nhiêu năm thì cộng dồn đủ 1 PB?

So the wallet alone fits a small hard drive in a year. Now do all four Trúc Xanh sources: wallet transactions, app logs, securities orders and prices by the minute. The code converts units

automatically and asks one more question: if data grows 1.5 times each year, after how many years does the running total reach 1 PB?

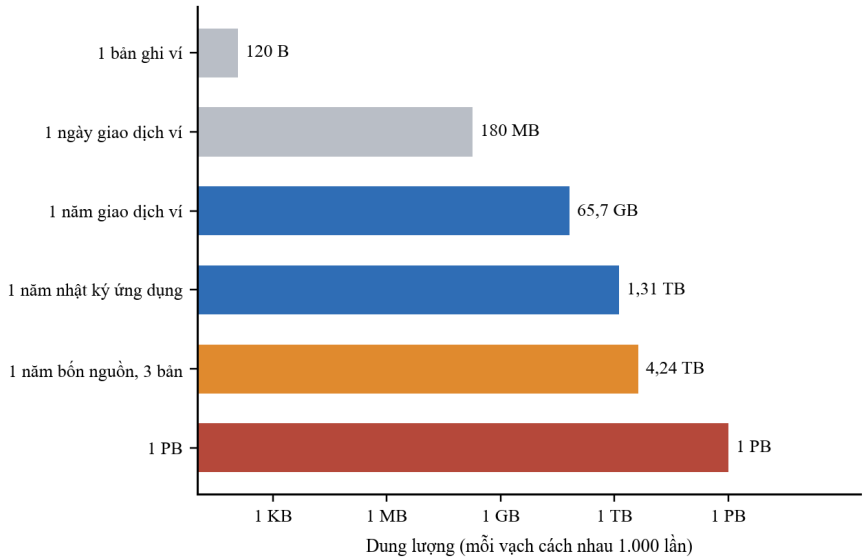
```
NGUON = [
    ("Giao dich vi", 1500000, 120, 365),
    ("Nhat ky ung dung", 12000000, 300, 365),
    ("Lenh chung khoan", 900000, 150, 250),
    ("Gia theo phut", 96000, 60, 250),
]
def don_vi(byte):
    for ten in ("B", "KB", "MB", "GB", "TB", "PB"):
        if byte < 1000 or ten == "PB":
            return f"{byte:,.2f} {ten}"
        byte /= 1000
tong = 0
for ten, n, b, t in NGUON:
    nam = n * b * t
    tong += nam
    print(f"{ten:<18}{don_vi(nam):>14}")
print(f"{'Tong':<18}{don_vi(tong):>14}")
print(f"{'Nhan ban 3':<18}{don_vi(tong * 3):>14}")
tich, nam, ca_nam = 0, 0, tong
while tich < 1e15:
    nam += 1
    tich += ca_nam
    ca_nam *= 1.5
print("Nam dat 1 PB:", nam)
```

Kết quả chạy / *Output*

```
Giao dich vi            65.70 GB
Nhat ky ung dung         1.31 TB
Lenh chung khoan        33.75 GB
Gia theo phut            1.44 GB
Tong                     1.41 TB
Nhan ban 3               4.24 TB
Nam dat 1 PB: 15
```

Nhật ký ứng dụng chiếm 1,31 TB, khoảng 92,9% tổng, dù mỗi sự kiện chỉ 300 byte. Lý do là có 12 triệu sự kiện mỗi ngày. Giá theo phút chỉ 1,44 GB: nhiều mã nhưng ít dòng. Tổng bốn nguồn là 1,41 TB một năm, nhân ba bản sao thành 4,24 TB. Nếu mỗi năm dữ liệu gấp 1,5 lần, phải đến năm thứ 15 mới cộng dồn đủ 1 PB. Ước tính này bỏ qua chỉ mục và nén, nên sai lệch đúng chiều của một phép ước lượng thô.

App logs take 1.31 TB, about 92.9% of the total, although each event is only 300 bytes. The cause is 12 million events a day. Prices by the minute take just 1.44 GB: many tickers but few rows. The four sources total 1.41 TB a year, or 4.24 TB with three copies. If data grows 1.5 times each year, the running total reaches 1 PB only in year 15. The estimate ignores indexes and compression, so treat it as a rough figure.



Hình 1.2. Mỗi thanh dài hơn thanh trước vài bậc, vì trục chia theo từng nấc 1.000 lần. Nhìn thanh cam và thanh đỏ: dữ liệu bốn nguồn nhân ba bản sao vẫn còn kém ngưỡng 1 PB gần 240 lần.

Figure 1.2. *Each bar is several steps longer than the one before, because the axis moves in jumps of 1,000. Compare the orange and red bars: four sources times three copies are still almost 240 times short of 1 PB.*

PHẢI THUỘC / MEMORISE – Dung lượng một năm $D = N * b * t$. Đổi byte lên đơn vị lớn bằng cách chia liên tiếp cho 1.000. Dung lượng thực chiếm = D nhân số bản sao. 1 byte = 8 bit.
Annual storage $D = N * b * t$. Move bytes up to larger units by dividing repeatedly by 1,000. Real storage = D times the number of copies. 1 byte = 8 bits.

$$D_{tổng} = D * r \tag{1.2}$$

LỖI HAY GẶP / COMMON MISTAKE – Nhầm bit với byte, hoặc nhầm 1.000 với 1.024 mà không ghi rõ. Sai số 2,4% ở KB thành khoảng 10% ở TB, đủ làm lệch ngân sách mua ổ cứng. Quên nhân bản sao là lỗi hay gặp thứ hai.
Mixing up bits and bytes, or 1,000 and 1,024, without saying which. A 2.4% gap at KB grows to about 10% at TB, enough to skew a hardware budget. Forgetting the copies is the second common slip.

1.3. Ba dạng dữ liệu: có cấu trúc, bán cấu trúc, phi cấu trúc – *1.3. Three forms of data: structured, semi-structured, unstructured*

Hình dung ba cuốn sổ. Cuốn thứ nhất kẻ sẵn ô, mỗi ô một thông tin, như sổ thu chi. Cuốn thứ hai là những phiếu ghi chú có nhãn như Ngày, Người, Việc, nhưng phiếu nào cũng có thể thêm nhãn riêng. Cuốn thứ ba là thư tay và ảnh chụp, không có ô, không có nhãn. Ba cuốn sổ đó là ba dạng dữ liệu.
Picture three notebooks. The first has printed cells, one fact per cell, like an expense ledger. The second holds labelled slips such as Date, Person, Task, yet any slip may add a label of its own.

The third is full of handwritten letters and photographs, with no cells and no labels. These notebooks are the three forms of data.

Dữ liệu có cấu trúc xếp thành hàng và cột với kiểu cố định, như bảng trong cơ sở dữ liệu bạn đã học. Dữ liệu bán cấu trúc có nhãn đi kèm giá trị nhưng mỗi bản ghi có thể có bộ nhãn khác nhau. Hai dạng quen thuộc là CSV (comma-separated values, giá trị cách nhau bằng dấu phẩy) cho bảng phẳng và JSON (JavaScript Object Notation, ký pháp đối tượng) cho bản ghi có nhãn. Dữ liệu phi cấu trúc là văn bản tự do, ảnh, âm thanh, video.

Structured data sits in rows and columns with fixed types, like the tables in the database course. Semi-structured data carries labels beside values, but each record may have a different set of labels. Two familiar formats are CSV (comma-separated values) for flat tables and JSON (JavaScript Object Notation) for labelled records. Unstructured data is free text, images, audio and video.

Dạng *Form*	**Đặc điểm** *Trait*	**Ví dụ ở Trúc Xanh** *Example at Trúc Xanh*	**Cách xử lý** *How it is handled*
Có cấu trúc *Structured*	Cột và kiểu cố định *Fixed columns and types*	Bảng giao dịch, bảng hợp đồng vay *Transaction table, loan contract table*	Truy vấn SQL (ngôn ngữ truy vấn có cấu trúc), tổng hợp ngay *SQL (Structured Query Language) queries, summarise at once*
Bán cấu trúc *Semi-structured*	Có nhãn, bộ nhãn thay đổi *Labelled, label set varies*	Nhật ký ứng dụng dạng JSON, nhật ký máy chủ *JSON app logs, server logs*	Đọc từng dòng, lấy khóa cần dùng *Read line by line, pick the keys needed*
Phi cấu trúc *Unstructured*	Không có khuôn sẵn *No ready template*	Email khiếu nại, ảnh giấy tờ, ghi âm tổng đài *Complaint emails, document photos, call recordings*	Chuyển thành con số rồi mới tính *Turn into numbers before computing*

Bảo Ngọc nhận từ anh Quân tám dòng nhật ký của ứng dụng Trúc Xanh Pay. Mỗi dòng ghi thời điểm, mã người dùng, sự kiện, và có thể thêm số tiền, loại máy hoặc ghi chú. Chị Thảo muốn biết mỗi sự kiện xảy ra mấy lần, tổng tiền chuyển là bao nhiêu và có bao nhiêu lần đăng nhập không ghi loại máy.

Mr Quân hands Bảo Ngọc eight lines of the Trúc Xanh Pay app log. Each line records the time, the user code and the event, and may add an amount, a device type or a note. Ms Thảo wants to know how often each event occurs, how much money was transferred and how many logins have no device type.

Dòng *Row*	**Giờ** *Time*	**Người** *User*	**Sự kiện** *Event*	**Trường thêm** *Extra field*
1	09:00:03	u17	Đăng nhập *Login*	Máy: android *Device: android*

Dòng *Row*	Giờ *Time*	Người *User*	Sự kiện *Event*	Trường thêm *Extra field*
2	09:00:21	u17	Xem số dư *View balance*	(không có *none)*
3	09:01:10	u17	Chuyển tiền *Transfer*	Tiền: 500.000 đồng *Amount: 500,000 dong*
4	09:02:45	u42	Đăng nhập *Login*	Máy: ios *Device: ios*
5	09:03:02	u42	Xem số dư *View balance*	(không có *none)*
6	09:03:30	u42	Chuyển tiền *Transfer*	Tiền: 2.000.000 đồng *Amount: 2,000,000 dong*
7	09:05:12	u17	Đăng xuất *Logout*	Ghi chú: hết phiên *Note: session ended*
8	09:06:00	u58	Đăng nhập *Login*	(không có *none)*

Làm tay theo bốn bước. Bước 1, đọc từng dòng như một bản ghi có nhãn. Bước 2, đếm theo sự kiện: đăng nhập 3, xem số dư 2, chuyển tiền 2, đăng xuất 1. Bước 3, cộng trường tiền, chỉ có ở dòng 3 và dòng 6: 500.000 + 2.000.000 = 2.500.000 đồng. Bước 4, xét các dòng đăng nhập: dòng 8 không có loại máy, nên 1 trên 3 lần đăng nhập thiếu thông tin. Đoạn mã sau làm đúng bốn bước ấy bằng thư viện json. Lệnh get cho phép hỏi một khóa mà không lỗi khi khóa vắng.

Work it by hand in four steps. Step 1, read each line as a labelled record. Step 2, count by event: login 3, view balance 2, transfer 2, logout 1. Step 3, add the amount field, present only in rows 3 and 6: 500,000 + 2,000,000 = 2,500,000 dong. Step 4, look at the login rows: row 8 has no device type, so 1 of 3 logins lacks that detail. The next code does exactly those four steps with the json library. The get call asks for a key without failing when the key is absent.

```
import json
LOG = """
{"t":"09:00:03","u":"u17","ev":"dang_nhap","may":"android"}
{"t":"09:00:21","u":"u17","ev":"xem_so_du"}
{"t":"09:01:10","u":"u17","ev":"chuyen_tien","tien":500000}
{"t":"09:02:45","u":"u42","ev":"dang_nhap","may":"ios"}
{"t":"09:03:02","u":"u42","ev":"xem_so_du"}
{"t":"09:03:30","u":"u42","ev":"chuyen_tien","tien":2000000}
{"t":"09:05:12","u":"u17","ev":"dang_xuat","ghi_chu":"het phien"}
{"t":"09:06:00","u":"u58","ev":"dang_nhap"}
""".strip().splitlines()
dem, tong, thieu = {}, 0, 0
for dong in LOG:
    b = json.loads(dong)
    dem[b["ev"]] = dem.get(b["ev"], 0) + 1
    tong += b.get("tien", 0)
    if b["ev"] == "dang_nhap" and "may" not in b:
        thieu += 1
print("so dong:", len(LOG))
print("dem theo su kien:")
for k, v in dem.items():
```

```
    print("  ", k, v)
print("tong tien chuyen:", tong)
print("dang nhap thieu loai may:", thieu)
```

Kết quả chạy / *Output*

```
so dong: 8
dem theo su kien:
   dang_nhap 3
   xem_so_du 2
   chuyen_tien 2
   dang_xuat 1
tong tien chuyen: 2500000
dang nhap thieu loai may: 1
```

Kết quả khớp phép tính tay: 8 dòng, 2.500.000 đồng, 1 lần đăng nhập thiếu loại máy. Với một tệp nhật ký 50 triệu dòng, đoạn mã không đổi, chỉ khác là vòng for đọc từng dòng từ tệp thay vì từ danh sách. Mục 1.11 làm đúng việc đó.

The result matches the hand sums: 8 lines, 2,500,000 dong, 1 login missing the device type. For a log file of 50 million lines the code would stay the same; only the for loop would read each line from a file instead of from a list. Section 1.11 does exactly that.

> **LỖI HAY GẶP / COMMON MISTAKE –** Viết mã giả định mọi dòng JSON đều có đủ khóa, rồi dừng ở dòng đầu tiên thiếu khóa. Với dữ liệu bán cấu trúc luôn dùng get hoặc kiểm tra khóa trước, và đếm riêng số dòng thiếu để báo lại.
> *Writing code that assumes every JSON line has every key, then crashing at the first line without one. With semi-structured data always use get or test the key first, and count the lines with gaps so you can report them.*

1.4. Các nguồn dữ liệu lớn trong tài chính – *1.4. Big data sources in finance*

Một tổ chức tài chính nhận dữ liệu từ nhiều cửa. Cửa quen nhất là giao dịch: mỗi lần tiền ra vào tài khoản hay mỗi lệnh mua bán cổ phiếu để lại một dòng. Cửa mới hơn gồm dữ liệu từ ứng dụng điện thoại, tin tức, mạng xã hội, ảnh chụp từ vệ tinh, tín hiệu định vị. Chị Thảo dặn Bảo Ngọc: mỗi con số đưa lên báo cáo phải trả lời được câu hỏi số này lấy từ đâu.

A financial organisation receives data through many doors. The most familiar is the transaction: each movement of money in an account, or each share order, leaves a row. Newer doors include phone app data, news, social media, satellite photographs and location signals. Ms Thảo tells Bảo Ngọc that every figure in a report must answer the question where it came from.

Bảng tiếp theo liệt kê tám nhóm nguồn. Cột tần suất cho biết dữ liệu mới đến theo nhịp nào, vì nhịp quyết định cách xử lý: theo lô một lần mỗi ngày hay theo luồng từng giây. API (application programming interface, giao diện lập trình ứng dụng) là cổng để chương trình này lấy dữ liệu từ chương trình khác.

The next table lists eight groups of sources. The frequency column shows the pace at which new data arrives, and pace decides the method: batch once a day or a stream every second. An API (application programming interface) is a doorway through which one program fetches data from another.

Nguồn *Source*	**Tần suất** *Frequency*	**Dùng để làm gì** *What it is used for*
Giao dịch tài khoản và hợp đồng vay *Account transactions and loan contracts*	Mỗi lần ghi sổ *Each posting*	Đối soát, sao kê, tính lãi, báo cáo *Reconciliation, statements, interest, reports*
Giao dịch thẻ và ví điện tử *Card and e-wallet payments*	Từng giây *Every second*	Phát hiện gian lận, phân tích chi tiêu *Fraud detection, spending analysis*
Thị trường tần suất cao (giá, lệnh, tick) *High-frequency market data (prices, orders, ticks)*	Mili giây *Milliseconds*	Giám sát giao dịch, thuật toán chia lệnh *Trade monitoring, order-slicing algorithms*
Ngân hàng số: nhật ký ứng dụng *Digital banking: app logs*	Từng sự kiện *Each event*	Hiểu hành vi khách, phát hiện đăng nhập lạ *Understand behaviour, spot odd logins*
Tin tức và mạng xã hội *News and social media*	Từng phút *Every minute*	Đo tâm lý thị trường, cảnh báo rủi ro danh tiếng *Gauge sentiment, warn of reputation risk*
Dữ liệu thay thế: định vị, ảnh vệ tinh, hóa đơn điện *Alternative data: location, satellite images, utility bills*	Ngày đến tháng *Daily to monthly*	Chấm điểm khách chưa có lịch sử tín dụng *Score customers with no credit history*
Dữ liệu mở: thống kê, giá, văn bản quy định *Open data: statistics, prices, regulations*	Tháng đến quý *Monthly to quarterly*	Làm nền so sánh và dự báo vĩ mô *A base for comparison and macro forecasts*
Dữ liệu giám sát: báo cáo gửi cơ quan quản lý *Supervisory data: reports sent to regulators*	Ngày đến quý *Daily to quarterly*	Đáp ứng yêu cầu báo cáo, tự kiểm tra tuân thủ *Meet reporting duties, self-check compliance*

Dữ liệu thay thế (alternative data) là dữ liệu nằm ngoài hồ sơ tài chính truyền thống. Một công ty cho vay có thể nhìn thấy khách trả tiền điện đều đặn dù khách chưa từng vay ngân hàng. Dữ liệu này mở thêm cơ hội, nhưng cũng đặt ra câu hỏi về sự đồng ý của người có dữ liệu và về chất lượng; hai câu hỏi ấy quay lại ở các chương 3 và 9. Dữ liệu cá nhân gồm cả tên, ảnh, email, số tài khoản, bài đăng và địa chỉ mạng của thiết bị. Dữ liệu cá nhân gồm cả tên, ảnh, email, số tài khoản, bài đăng và địa chỉ mạng của thiết bị. Dữ liệu cá nhân gồm cả tên, ảnh, email, số tài khoản, bài đăng và địa chỉ mạng của thiết bị. Dữ liệu cá nhân gồm cả tên, ảnh, email, số tài khoản, bài đăng và địa chỉ mạng của thiết bị. Dữ liệu cá nhân gồm cả tên, ảnh, email, số tài khoản, bài đăng và địa chỉ mạng của thiết bị.

Alternative data is data from outside traditional financial records. A lender may see that a customer pays the electricity bill on time although that person has never borrowed from a bank. Such data opens opportunities, yet it raises questions about the consent of the person it describes and about its quality; both questions return in chapters 3 and 9. Personal data covers names, photos, emails, account numbers, posts and the network address of a device. Personal data covers names, photos, emails, account numbers, posts and the network address of a device. Personal data covers names, photos, emails, account numbers, posts and the network address of a device. Personal data covers names, photos, emails, account numbers, posts and the network address of a device. Personal data covers names, photos, emails, account numbers, posts and the network address of a device. Personal data covers names, photos, emails, account numbers, posts and the network address of a device.

> **LỖI HAY GẶP / COMMON MISTAKE –** Dùng dữ liệu cá nhân cho mục đích mới khi khách chưa đồng ý. Ví dụ lấy nhật ký ứng dụng để chấm điểm tín dụng trong khi điều khoản chỉ nói về bảo mật tài khoản. Phải kiểm mục đích đã công bố trước khi nối nguồn mới.
> *Using personal data for a new purpose without the customer's consent. For example taking the app log to score credit when the terms only mention account security. Check the declared purpose before joining a new source.*

1.5. Trí tuệ nhân tạo, học máy, học sâu và dữ liệu lớn – *1.5. Artificial intelligence, machine learning, deep learning and big data*

Bốn thuật ngữ này hay bị dùng lẫn. Hãy nghĩ tới một nhân viên mới ở phòng thẩm định cho vay. Nếu nhân viên chỉ làm theo cuốn sổ luật có sẵn, đó là một hệ thống AI kiểu luật viết tay. Nếu nhân viên tự rút kinh nghiệm từ hàng nghìn hồ sơ cũ, đó là học máy. Nếu nhân viên có nhiều tầng bộ lọc nối nhau, tầng đầu nhìn nét chữ, tầng sau nhìn từ, tầng cuối nhìn ý, đó là học sâu. Còn hồ sơ cũ chất đống trong kho là dữ liệu lớn.

These four terms get mixed up often. Think of a new clerk in the loan appraisal department. A clerk who only follows the ready rule book is an AI system of the hand-written rule kind. A clerk who draws experience from thousands of old files is doing machine learning. A clerk with several filters in a chain, the first looking at handwriting, the next at words, the last at meaning, is doing deep learning. The old files piled in the store are the big data.

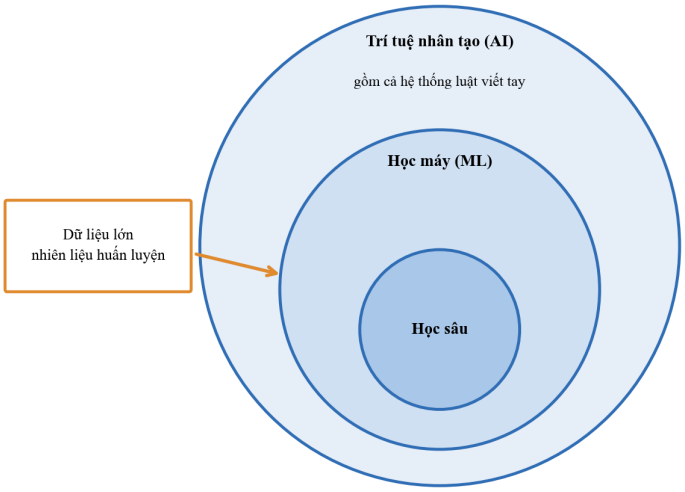
Theo OECD (2021), một hệ thống trí tuệ nhân tạo (AI, artificial intelligence) là hệ thống máy móc, với mục tiêu do con người đặt, đưa ra dự đoán, gợi ý hoặc quyết định. Hệ thống làm ba việc: nhận biết môi trường qua dữ liệu, gom cái nhận biết thành mô hình, rồi dùng mô hình suy ra lựa chọn. Mức tự chủ của các hệ thống khác nhau; có hệ thống chỉ gợi ý, có hệ thống tự hành động.

According to OECD (2021), an artificial intelligence (AI) system is a machine-based system that, for goals set by people, makes predictions, recommendations or decisions. It does three things: it senses its environment through data, it gathers what it senses into a model, and it uses the model to infer choices. Systems differ in autonomy; some only advise and some act alone.

Học máy (ML, machine learning) là phần của AI tự rút quy luật từ dữ liệu, không cần người viết từng luật. Học sâu (deep learning) là một nhánh của học máy dùng mạng nơ-

ron nhiều tầng, hợp với dữ liệu phi cấu trúc như ảnh và giọng nói. Hình 1.3 vẽ ba vòng lồng nhau, và dữ liệu lớn đứng ngoài, nuôi các mô hình.

Machine learning (ML) is the part of AI that draws patterns from data without a person writing every rule. Deep learning is a branch of machine learning that uses neural networks with many layers, suited to unstructured data such as images and speech. Figure 1.3 draws three nested rings, with big data outside, feeding the models.



Hình 1.3. Vòng ngoài là AI, vòng giữa là học máy, vòng trong là học sâu. Hộp cam bên trái là dữ liệu lớn: mũi tên cho thấy nó đi vào học máy từ bên ngoài ba vòng.

Figure 1.3. *The outer ring is AI, the middle ring machine learning, the inner ring deep learning. The orange box on the left is big data: the arrow shows it feeding machine learning and sitting outside the three rings.*

Có ba cách học chính. Học có giám sát (supervised learning) dùng ví dụ đã có đáp án: hồ sơ vay cũ kèm nhãn trả đúng hạn hoặc trễ hạn, để dự đoán hồ sơ mới. Học không giám sát (unsupervised learning) tìm nhóm trong dữ liệu chưa có nhãn, như chia khách thành các nhóm chi tiêu giống nhau. Học tăng cường (reinforcement learning) học qua thử và nhận thưởng hoặc phạt, như thuật toán chia lệnh thử nhiều nhịp bán để chọn nhịp cho giá tốt.

There are three main ways of learning. Supervised learning uses examples that already carry answers: old loan files labelled repaid on time or late, to predict a new file. Unsupervised learning finds groups in unlabelled data, such as splitting customers into groups with similar spending. Reinforcement learning learns by trial with rewards and penalties, as when an order-slicing algorithm tries several selling paces to find the one that earns a better price.

Một hệ thống AI trải qua bốn giai đoạn trong vòng đời: thiết kế, thu thập và xử lý dữ liệu rồi dựng mô hình; kiểm tra và thẩm định; triển khai; vận hành và giám sát. Học máy thường tốt hơn khi được huấn luyện bằng nhiều ví dụ hơn, nên giai đoạn thu thập dữ liệu tốn nhiều công sức. AI cũng có mặt ngoài tài chính, như y tế, ô tô và các thiết bị kết nối mạng.

An AI system passes through four stages in its life: design, collecting and processing data and then building the model; testing and validation; deployment; operation and monitoring. Machine

learning usually improves when it trains on more examples, so the data collection stage takes much effort. AI also appears outside finance, in health care, cars and connected devices.

Một hệ thống AI trải qua bốn giai đoạn trong vòng đời: thiết kế, thu thập và xử lý dữ liệu rồi dựng mô hình; kiểm tra và thẩm định; triển khai; vận hành và giám sát. Học máy thường tốt hơn khi được huấn luyện bằng nhiều ví dụ hơn, nên giai đoạn thu thập dữ liệu tốn nhiều công sức. AI cũng có mặt ngoài tài chính, như y tế, ô tô và các thiết bị kết nối mạng.

An AI system passes through four stages in its life: design, collecting and processing data and then building the model; testing and validation; deployment; operation and monitoring. Machine learning usually improves when it trains on more examples, so the data collection stage takes much effort. AI also appears outside finance, in health care, cars and connected devices.

Một hệ thống AI trải qua bốn giai đoạn trong vòng đời: thiết kế, thu thập và xử lý dữ liệu rồi dựng mô hình; kiểm tra và thẩm định; triển khai; vận hành và giám sát. Học máy thường tốt hơn khi được huấn luyện bằng nhiều ví dụ hơn, nên giai đoạn thu thập dữ liệu tốn nhiều công sức. AI cũng có mặt ngoài tài chính, như y tế, ô tô và các thiết bị kết nối mạng.

An AI system passes through four stages in its life: design, collecting and processing data and then building the model; testing and validation; deployment; operation and monitoring. Machine learning usually improves when it trains on more examples, so the data collection stage takes much effort. AI also appears outside finance, in health care, cars and connected devices.

Một hệ thống AI trải qua bốn giai đoạn trong vòng đời: thiết kế, thu thập và xử lý dữ liệu rồi dựng mô hình; kiểm tra và thẩm định; triển khai; vận hành và giám sát. Học máy thường tốt hơn khi được huấn luyện bằng nhiều ví dụ hơn, nên giai đoạn thu thập dữ liệu tốn nhiều công sức. AI cũng có mặt ngoài tài chính, như y tế, ô tô và các thiết bị kết nối mạng.

An AI system passes through four stages in its life: design, collecting and processing data and then building the model; testing and validation; deployment; operation and monitoring. Machine learning usually improves when it trains on more examples, so the data collection stage takes much effort. AI also appears outside finance, in health care, cars and connected devices.

Một hệ thống AI trải qua bốn giai đoạn trong vòng đời: thiết kế, thu thập và xử lý dữ liệu rồi dựng mô hình; kiểm tra và thẩm định; triển khai; vận hành và giám sát. Học máy thường tốt hơn khi được huấn luyện bằng nhiều ví dụ hơn, nên giai đoạn thu thập dữ liệu tốn nhiều công sức. AI cũng có mặt ngoài tài chính, như y tế, ô tô và các thiết bị kết nối mạng.

An AI system passes through four stages in its life: design, collecting and processing data and then building the model; testing and validation; deployment; operation and monitoring. Machine learning usually improves when it trains on more examples, so the data collection stage takes much effort. AI also appears outside finance, in health care, cars and connected devices.

Một hệ thống AI trải qua bốn giai đoạn trong vòng đời: thiết kế, thu thập và xử lý dữ liệu rồi dựng mô hình; kiểm tra và thẩm định; triển khai; vận hành và giám sát. Học máy thường tốt hơn khi được huấn luyện bằng nhiều ví dụ hơn, nên giai đoạn thu thập dữ liệu

tốn nhiều công sức. AI cũng có mặt ngoài tài chính, như y tế, ô tô và các thiết bị kết nối mạng.

An AI system passes through four stages in its life: design, collecting and processing data and then building the model; testing and validation; deployment; operation and monitoring. Machine learning usually improves when it trains on more examples, so the data collection stage takes much effort. AI also appears outside finance, in health care, cars and connected devices.

Khái niệm *Concept*	**Là gì** *What it is*	**Ví dụ ở Trúc Xanh** *Example at Trúc Xanh*	**Cần dữ liệu lớn?** *Needs big data?*
AI *AI*	Máy đưa ra dự đoán, gợi ý, quyết định *Machine that predicts, advises, decides*	Bộ luật chặn thẻ viết tay *A hand-written card-blocking rule set*	Không bắt buộc *Not necessarily*
Học máy *Machine learning*	Máy rút quy luật từ dữ liệu *Machine draws patterns from data*	Mô hình dự đoán trễ hạn từ hồ sơ cũ *Model predicting late repayment from old files*	Thường cần *Usually*
Học sâu *Deep learning*	Học máy dùng mạng nơ-ron nhiều tầng *Machine learning with many-layered networks*	Đọc chữ trên ảnh giấy tờ *Reading text on document photos*	Cần nhiều *Heavily*
Dữ liệu lớn *Big data*	Nguồn dữ liệu để huấn luyện và kiểm tra *Source data for training and testing*	Nhật ký, giao dịch, ảnh nhiều năm *Years of logs, transactions, photos*	Chính nó *It is the data*

Bài mẫu phân loại bốn tình huống ở Trúc Xanh. Tình huống một: dự đoán khách trễ hạn bằng 10.000 hồ sơ đã gắn nhãn. Có nhãn nên là học có giám sát. Tình huống hai: chia 50.000 khách thành nhóm chi tiêu, chưa có nhãn nào, nên là học không giám sát. Tình huống ba: thuật toán chia lệnh tự thử các nhịp bán rồi nhận thưởng khi giá khớp tốt, nên là học tăng cường. Tình huống bốn: đọc chữ trên ảnh giấy tờ, dữ liệu phi cấu trúc và mạng nhiều tầng, nên là học sâu, thường cũng theo cách có giám sát.

A worked example sorts four situations at Trúc Xanh. Situation one: predict late customers from 10,000 labelled files. Labels exist, so it is supervised learning. Situation two: split 50,000 customers into spending groups with no label at all, so it is unsupervised learning. Situation three: an order-slicing algorithm tries selling paces itself and gets rewarded when the price is good, so it is reinforcement learning. Situation four: reading text on document photos, unstructured data and a many-layered network, so it is deep learning, usually also in the supervised way.

> **PHẢI THUỘC / MEMORISE –** AI chứa học máy, học máy chứa học sâu. Dữ liệu lớn là nguyên liệu nằm ngoài ba vòng của AI. Có nhãn: học có giám sát. Chưa nhãn, tìm nhóm: học không giám sát. Thử và nhận thưởng: học tăng cường.
> *AI contains machine learning, and machine learning contains deep learning. Big data is the raw*

> *material outside the three rings of AI. With labels: supervised. Without labels, finding groups: unsupervised. Trial and reward: reinforcement.*

1.6. Vì sao AI cần dữ liệu lớn, và dữ liệu lớn cần AI – *1.6. Why AI needs big data, and big data needs AI*

Quan hệ này đi hai chiều. Chiều thứ nhất: AI cần dữ liệu lớn. Một mô hình học máy giống người học nghề, xem càng nhiều ca càng nhận ra quy luật, nhất là ca hiếm. Gian lận thẻ là ca hiếm. Giả sử cứ 1.000 giao dịch có 2 giao dịch gian lận, tức 0,2%. Mô hình cần thấy ít nhất vài trăm ca gian lận mới học được dấu hiệu. Hãy tính cần bao nhiêu giao dịch.

The relation runs both ways. First, AI needs big data. A machine learning model is like an apprentice: the more cases it sees, the better it spots patterns, above all in rare cases. Card fraud is a rare case. Suppose 2 in every 1,000 transactions are fraud, which is 0.2%. The model must see at least a few hundred fraud cases to learn the signs. Work out how many transactions that takes.

1. Chọn số ca gian lận tối thiểu mô hình cần thấy: 400.
   Choose the minimum number of fraud cases the model must see: 400.
2. Chia cho tỷ lệ gian lận: 400 ÷ 0,002 = 200.000 giao dịch.
   Divide by the fraud rate: 400 ÷ 0.002 = 200,000 transactions.
3. So với nguồn có sẵn: ví Trúc Xanh Pay ghi 1.500.000 giao dịch mỗi ngày, nên 200.000 giao dịch chỉ bằng khoảng 3,2 giờ dữ liệu.
   Compare with the supply: the Trúc Xanh Pay wallet records 1,500,000 transactions a day, so 200,000 transactions are only about 3.2 hours of data.

Nghĩa là muốn học ca hiếm thì cần dữ liệu từ nhiều ngày và nhiều tháng, không thể chỉ vài trăm dòng. Tỷ lệ ca hiếm càng thấp, lượng dữ liệu cần càng lớn.

So learning a rare case needs data from many days and months, not a few hundred rows. The rarer the case, the more data it takes.

Chiều thứ hai: dữ liệu lớn cần AI. Giả sử một nhân viên xem mỗi sự kiện nhật ký mất 10 giây và làm 8 giờ mỗi ngày. Số sự kiện một người xem được là 8 × 3.600 ÷ 10 = 2.880. Nhật ký ứng dụng có 12.000.000 sự kiện mỗi ngày, nên cần 12.000.000 ÷ 2.880 ≈ 4.167 người chỉ để xem một lượt. Không ngân hàng nào làm vậy. Máy phải lọc trước, và người chỉ xem phần máy gắn cờ.

The second direction: big data needs AI. Suppose an employee takes 10 seconds to look at one log event and works 8 hours a day. One person can view 8 × 3,600 ÷ 10 = 2,880 events. The app log holds 12,000,000 events a day, so it would take 12,000,000 ÷ 2,880 ≈ 4,167 people merely to glance at each once. No bank does that. Machines must filter first, and people review only what the machine flags.

Đại dịch COVID-19 đã đẩy nhanh quá trình số hóa vốn có từ trước, kể cả việc dùng AI. Hai điều kiện khác mở đường cho AI lan vào tài chính: dữ liệu dồi dào và năng lực tính toán rẻ hơn. Năng lực tính toán ở đây là CPU (bộ xử lý trung tâm, central processing unit) và GPU (bộ xử lý đồ họa, graphics processing unit); GPU làm được hàng nghìn

phép tính nhỏ cùng lúc nên hợp với huấn luyện mô hình. Điện toán đám mây (cloud computing) cho phép thuê máy theo giờ thay vì mua cả phòng máy, nên một nhóm nhỏ cũng dựng được hệ thống lớn. Cái giá phải trả là sự phụ thuộc vào vài nhà cung cấp; chương 2 và chương 9 quay lại việc này.

The COVID-19 pandemic sped up a digitalisation trend that already existed, including the use of AI. Two further conditions opened the way for AI in finance: plentiful data and cheaper computing power. Computing power here means the CPU (central processing unit) and the GPU (graphics processing unit); a GPU performs thousands of small calculations at once, which suits model training. Cloud computing lets a team rent machines by the hour instead of buying a whole machine room, so a small team can build a large system. The price is dependence on a few providers; chapters 2 and 9 come back to this.

Mức dùng AI và dữ liệu lớn trong tài chính đang tăng ở ba mặt. Về nghiên cứu, số công trình về AI tăng nhanh hơn số công trình tin học nói chung. Về đầu tư, chi tiêu cho AI và vốn rót vào các công ty khởi nghiệp AI cùng tăng. Về sản phẩm, ngân hàng bán lẻ dùng AI để cá nhân hóa sản phẩm, trả lời khách bằng chương trình trò chuyện, chấm điểm tín dụng, dự báo tổn thất cho vay, phát hiện gian lận và chống rửa tiền (AML, anti-money laundering). Quản lý tài sản dùng AI để tư vấn tự động và quản lý rủi ro. Bảo hiểm dùng AI để xử lý hồ sơ bồi thường.

The use of AI and big data in finance is rising on three fronts. In research, the volume of AI papers grows faster than computer science output overall. In investment, spending on AI and money flowing to AI start-ups both rise. In products, retail banks use AI to tailor products, answer customers through chat programs, score credit, forecast loan losses, detect fraud and fight money laundering (AML, anti-money laundering). Asset managers use AI for automated advice and risk management. Insurers use AI to handle claims.

1.7. Lợi ích và rủi ro tổng quan – *1.7. Benefits and risks at a glance*

Giá trị mà AI và dữ liệu lớn mang lại đi theo hai hướng. Hướng thứ nhất là hiệu quả của chính tổ chức: giảm chi phí, tự động hóa việc lặp lại, ra quyết định nhanh hơn, kiểm soát rủi ro và tuân thủ tốt hơn. Hướng thứ hai là chất lượng dịch vụ cho khách: sản phẩm hợp từng người hơn, nhiều lựa chọn hơn, đôi khi giá thấp hơn. Khách hàng hưởng lợi khi tổ chức cạnh tranh bằng hai hướng ấy.

The value AI and big data bring runs in two directions. The first is the organisation's own efficiency: lower cost, automation of repeated work, faster decisions, and better control of risk and compliance. The second is service quality for customers: products that fit each person better, more choice, sometimes lower prices. Customers gain when organisations compete along both directions.

Mỗi lợi ích đi kèm một rủi ro. Bảng sau ghép từng cặp và chỉ chương trong sách sẽ đi sâu. Đọc theo hàng ngang: cột giữa là ví dụ, cột phải là cái giá phải tính trước.

Each benefit comes with a risk. The next table pairs them and points to the chapter that goes deeper. Read along each row: the middle column is an example, the right-hand column is the price to count in advance.

Lợi ích *Benefit*	**Ví dụ** *Example*	**Rủi ro đi kèm** *Attached risk*	**Chương** *Chapter*
Giảm chi phí, tự động hóa *Lower cost, automation*	Máy đối soát hàng triệu dòng mỗi đêm *A machine reconciles millions of rows nightly*	Gián đoạn dịch vụ khi hệ thống ngừng; cần người thay thế *Service disruption when the system stops; people must stand in*	5, 9
Sản phẩm hợp từng khách *Products that fit each customer*	Gợi ý gói vay theo thói quen chi tiêu *Suggesting a loan package from spending habits*	Thiên lệch, đối xử không công bằng *Bias and unfair treatment*	7, 9
Mở tín dụng cho người chưa có lịch sử *Credit for people with no history*	Chấm điểm bằng giao dịch ví *Scoring with wallet transactions*	Biến đại diện gây phân biệt; khó giải thích lý do từ chối *Proxy variables that discriminate; hard to explain a refusal*	7, 9
Thực hiện lệnh tốt hơn *Better trade execution*	Chia lệnh lớn theo nhịp thị trường *Slicing a large order to the market's pace*	Hành vi bầy đàn, sập chớp nhoáng, thao túng *Herding, flash crashes, manipulation*	6
Giám sát và tuân thủ *Oversight and compliance*	Lọc giao dịch nghi ngờ rửa tiền *Filtering suspected money-laundering transactions*	Cảnh báo sai nhiều; mô hình khó kiểm toán *Many false alerts; models hard to audit*	4, 8
Quản lý rủi ro *Risk management*	Dự báo tổn thất cho vay *Forecasting loan losses*	Mô hình trôi, hỏng khi có cú sốc *Model drift, failure in a shock*	5

Ngoài các cặp trên còn có những rủi ro chung. Dữ liệu kém chất lượng đẻ ra mô hình sai, và dữ liệu cá nhân bị lộ gây hại cho khách. Vài nhà cung cấp đám mây hay mô hình có thể chiếm phần lớn thị trường, nên một sự cố ở họ lan ra nhiều tổ chức. Nhiều công ty dùng cùng một loại mô hình có thể cùng bán một lúc khi thị trường căng thẳng. Khi mô hình sai, câu hỏi ai chịu trách nhiệm phải có lời đáp rõ ràng. Các cơ quan quản lý vì vậy muốn tổ chức công bố việc dùng AI cho khách, tách người thẩm định mô hình khỏi người dựng mô hình, và kiểm tra mô hình trong điều kiện thị trường cực đoan.

Beyond these pairs sit shared risks. Poor data breeds wrong models, and leaked personal data harms customers. A few cloud or model providers may hold most of the market, so a fault at them spreads to many organisations. Many firms using the same type of model may all sell at once when markets are tense. When a model errs, the question of who is accountable needs a clear answer. Regulators therefore want organisations to tell customers when AI is used, to keep model validators apart from model builders, and to test models under extreme market conditions.

Họ cũng cân nhắc mức quản lý theo mức độ quan trọng của ứng dụng: ứng dụng quyết định cho vay cần chặt hơn ứng dụng gợi ý bài viết. Mục tiêu là vừa không chặn đổi mới, vừa giữ an toàn cho khách và thị trường. Quy định hiện hành có thể chưa khớp với AI, và

mỗi nước quản lý một kiểu, nên khung pháp lý dễ bị phân mảnh. Công tắc ngắt khẩn cấp (kill switch) tắt được mô hình lỗi, nhưng chính việc tắt cũng gây rủi ro vận hành. Công ty công nghệ lớn (BigTech) giữ lượng dữ liệu khách khổng lồ, nên đặt ra câu hỏi về cạnh tranh. Trong tài chính dựa trên chuỗi khối, AI có thể giúp hợp đồng thông minh tự điều chỉnh, nhưng dữ liệu đầu vào kém thì kết quả vẫn kém. Quy định hiện hành có thể chưa khớp với AI, và mỗi nước quản lý một kiểu, nên khung pháp lý dễ bị phân mảnh. Công tắc ngắt khẩn cấp (kill switch) tắt được mô hình lỗi, nhưng chính việc tắt cũng gây rủi ro vận hành. Công ty công nghệ lớn (BigTech) giữ lượng dữ liệu khách khổng lồ, nên đặt ra câu hỏi về cạnh tranh. Trong tài chính dựa trên chuỗi khối, AI có thể giúp hợp đồng thông minh tự điều chỉnh, nhưng dữ liệu đầu vào kém thì kết quả vẫn kém. Quy định hiện hành có thể chưa khớp với AI, và mỗi nước quản lý một kiểu, nên khung pháp lý dễ bị phân mảnh. Công tắc ngắt khẩn cấp (kill switch) tắt được mô hình lỗi, nhưng chính việc tắt cũng gây rủi ro vận hành. Công ty công nghệ lớn (BigTech) giữ lượng dữ liệu khách khổng lồ, nên đặt ra câu hỏi về cạnh tranh. Trong tài chính dựa trên chuỗi khối, AI có thể giúp hợp đồng thông minh tự điều chỉnh, nhưng dữ liệu đầu vào kém thì kết quả vẫn kém. Quy định hiện hành có thể chưa khớp với AI, và mỗi nước quản lý một kiểu, nên khung pháp lý dễ bị phân mảnh. Công tắc ngắt khẩn cấp (kill switch) tắt được mô hình lỗi, nhưng chính việc tắt cũng gây rủi ro vận hành. Công ty công nghệ lớn (BigTech) giữ lượng dữ liệu khách khổng lồ, nên đặt ra câu hỏi về cạnh tranh. Trong tài chính dựa trên chuỗi khối, AI có thể giúp hợp đồng thông minh tự điều chỉnh, nhưng dữ liệu đầu vào kém thì kết quả vẫn kém. Quy định hiện hành có thể chưa khớp với AI, và mỗi nước quản lý một kiểu, nên khung pháp lý dễ bị phân mảnh. Công tắc ngắt khẩn cấp (kill switch) tắt được mô hình lỗi, nhưng chính việc tắt cũng gây rủi ro vận hành. Công ty công nghệ lớn (BigTech) giữ lượng dữ liệu khách khổng lồ, nên đặt ra câu hỏi về cạnh tranh. Trong tài chính dựa trên chuỗi khối, AI có thể giúp hợp đồng thông minh tự điều chỉnh, nhưng dữ liệu đầu vào kém thì kết quả vẫn kém. Quy định hiện hành có thể chưa khớp với AI, và mỗi nước quản lý một kiểu, nên khung pháp lý dễ bị phân mảnh. Công tắc ngắt khẩn cấp (kill switch) tắt được mô hình lỗi, nhưng chính việc tắt cũng gây rủi ro vận hành. Công ty công nghệ lớn (BigTech) giữ lượng dữ liệu khách khổng lồ, nên đặt ra câu hỏi về cạnh tranh. Trong tài chính dựa trên chuỗi khối, AI có thể giúp hợp đồng thông minh tự điều chỉnh, nhưng dữ liệu đầu vào kém thì kết quả vẫn kém. Các chương 5 đến 9 triển khai từng chủ đề.

They also weigh the level of control by how much an application matters: a lending decision needs tighter control than an article recommender. The aim is to avoid blocking innovation while keeping customers and markets safe. Existing rules may not fit AI, and each country regulates in its own way, so the legal framework tends to fragment. A kill switch can shut down a faulty model, yet shutting down itself creates operational risk. Large technology firms (BigTech) hold vast stores of customer data, which raises questions about competition. In finance built on blockchains, AI may help smart contracts adjust themselves, but poor input data still gives poor output. Existing rules may not fit AI, and each country regulates in its own way, so the legal framework tends to fragment. A kill switch can shut down a faulty model, yet shutting down itself creates operational risk. Large technology firms (BigTech) hold vast stores of customer data,

which raises questions about competition. In finance built on blockchains, AI may help smart contracts adjust themselves, but poor input data still gives poor output. Existing rules may not fit AI, and each country regulates in its own way, so the legal framework tends to fragment. A kill switch can shut down a faulty model, yet shutting down itself creates operational risk. Large technology firms (BigTech) hold vast stores of customer data, which raises questions about competition. In finance built on blockchains, AI may help smart contracts adjust themselves, but poor input data still gives poor output. Existing rules may not fit AI, and each country regulates in its own way, so the legal framework tends to fragment. A kill switch can shut down a faulty model, yet shutting down itself creates operational risk. Large technology firms (BigTech) hold vast stores of customer data, which raises questions about competition. In finance built on blockchains, AI may help smart contracts adjust themselves, but poor input data still gives poor output. Existing rules may not fit AI, and each country regulates in its own way, so the legal framework tends to fragment. A kill switch can shut down a faulty model, yet shutting down itself creates operational risk. Large technology firms (BigTech) hold vast stores of customer data, which raises questions about competition. In finance built on blockchains, AI may help smart contracts adjust themselves, but poor input data still gives poor output. Existing rules may not fit AI, and each country regulates in its own way, so the legal framework tends to fragment. A kill switch can shut down a faulty model, yet shutting down itself creates operational risk. Large technology firms (BigTech) hold vast stores of customer data, which raises questions about competition. In finance built on blockchains, AI may help smart contracts adjust themselves, but poor input data still gives poor output. Chapters 5 to 9 develop each theme.

Lập luận ủng hộ và phản đối: để mô hình tự duyệt khoản vay nhỏ – ***For and against: letting a model approve small loans on its own***

Ủng hộ *For*	**Phản đối** *Against*
Duyệt trong vài giây, khách không phải chờ nhiều ngày. *Approval in seconds, with no wait of several days.*	Khoản vay ảnh hưởng lớn đến đời sống, nên cần người xem. *A loan changes a person's life, so a person should look.*
Mọi hồ sơ được xét theo cùng một thước đo. *Every file is judged by the same yardstick.*	Thiên lệch ẩn trong dữ liệu khó nhận ra khi máy tự quyết. *Bias hidden in the data is hard to spot when the machine decides alone.*
Chi phí thấp, mở được hạn mức nhỏ cho nhiều khách. *Low cost, so small limits can be opened for many customers.*	Khách bị từ chối khó biết lý do để khiếu nại. *A refused customer can hardly learn the reason to appeal.*

Cách dung hòa thường gặp là con người trong vòng lặp (human in the loop): máy xử lý hồ sơ rõ ràng, người xem hồ sơ sát ngưỡng và hồ sơ lớn. Khách cũng cần có đường phản đối kết quả do mô hình đưa ra và nhờ một người xem xét lại. Khách cũng cần có đường phản đối kết quả do mô hình đưa ra và nhờ một người xem xét lại. Khách cũng cần có đường phản đối kết quả do mô hình đưa ra và nhờ một người xem xét lại. Khách cũng cần có đường phản đối kết quả do mô hình đưa ra và nhờ một người xem xét lại. Khách

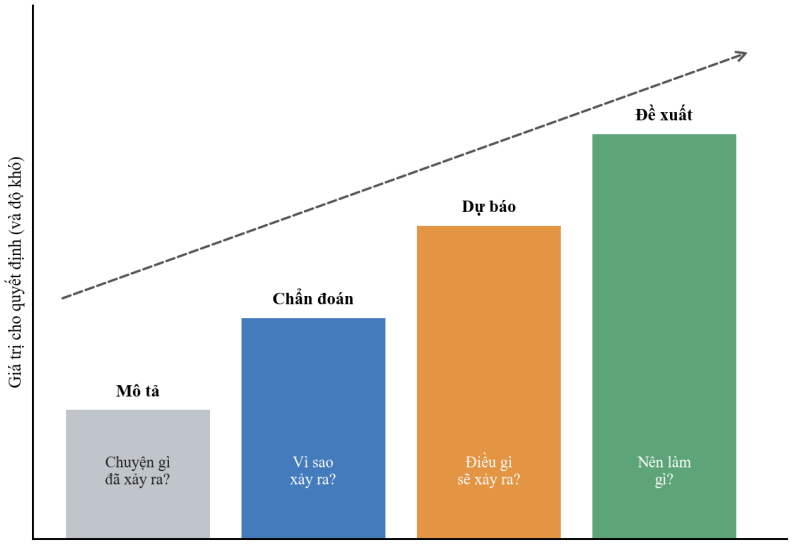
cũng cần có đường phản đối kết quả do mô hình đưa ra và nhờ một người xem xét lại. Khách cũng cần có đường phản đối kết quả do mô hình đưa ra và nhờ một người xem xét lại.

A common compromise is a person in the loop: the machine handles clear-cut files and a person reviews borderline and large ones. Customers also need a route to contest a model's outcome and ask a person to look again. Customers also need a route to contest a model's outcome and ask a person to look again. Customers also need a route to contest a model's outcome and ask a person to look again. Customers also need a route to contest a model's outcome and ask a person to look again. Customers also need a route to contest a model's outcome and ask a person to look again. Customers also need a route to contest a model's outcome and ask a person to look again.

1.8. Bốn loại phân tích – *1.8. Four kinds of analytics*

Chị Thảo thường hỏi Bảo Ngọc bốn kiểu câu. Chuyện gì đã xảy ra? Vì sao xảy ra? Điều gì sẽ xảy ra? Nên làm gì? Mỗi kiểu ứng với một loại phân tích, và đi từ trái sang phải thì giá trị cho quyết định tăng, độ khó cũng tăng. Hình 1.4 vẽ bốn bậc.

Ms Thảo asks Bảo Ngọc four kinds of question. What happened? Why did it happen? What will happen? What should we do? Each kind matches one type of analytics, and from left to right both the value for decisions and the difficulty rise. Figure 1.4 draws the four steps.



Hình 1.4. Bốn cột là bốn loại phân tích, cột cao hơn cho giá trị quyết định lớn hơn và khó làm hơn. Câu hỏi nằm trong mỗi cột; mũi tên đứt nét chỉ hướng đi lên.

Figure 1.4. *The four columns are the four kinds of analytics; a taller column means more decision value and more difficulty. The question sits inside each column; the dashed arrow points the way up.*

Loại *Kind*	**Câu hỏi** *Question*	**Ví dụ ở Trúc Xanh** *Example at Trúc Xanh*	**Kỹ thuật điển hình** *Typical technique*
Mô tả *Descriptive*	Chuyện gì đã xảy ra? *What happened?*	Tổng giao dịch ví mỗi tháng *Total wallet transactions per month*	Tổng, trung bình, đếm theo nhóm *Sums, averages, counts by group*

Loại *Kind*	**Câu hỏi** *Question*	**Ví dụ ở Trúc Xanh** *Example at Trúc Xanh*	**Kỹ thuật điển hình** *Typical technique*
Chẩn đoán *Diagnostic*	Vì sao xảy ra? *Why did it happen?*	Tháng 6 tăng vì kênh nào *Which channel drove June's rise*	So sánh nhóm, tách theo kênh *Compare groups, split by channel*
Dự báo *Predictive*	Điều gì sẽ xảy ra? *What will happen?*	Giao dịch tháng 7 *July transactions*	Xu hướng, hồi quy, học máy *Trend, regression, machine learning*
Đề xuất *Prescriptive*	Nên làm gì? *What should we do?*	Có mở rộng công suất không *Whether to expand capacity*	Quy tắc quyết định, tối ưu hóa *Decision rules, optimisation*

Làm tay với số giao dịch ví sáu tháng đầu năm, đơn vị nghìn giao dịch: 410, 435, 460, 520, 580, 640. Tháng 5 gồm 340 qua mã QR (quick response, phản hồi nhanh) và 240 qua ứng dụng; tháng 6 gồm 410 qua mã QR và 230 qua ứng dụng. Công suất xử lý của hệ thống là 750 nghìn giao dịch mỗi tháng, và nhóm kỹ thuật muốn mở rộng khi dự báo vượt 90% công suất.

Work by hand with wallet transactions for the first six months, in thousands: 410, 435, 460, 520, 580, 640. May has 340 through QR code (quick response) and 240 through the app; June has 410 through QR code and 230 through the app. The system can process 750 thousand transactions a month, and the engineering team wants to expand when the forecast passes 90% of capacity.

1. Mô tả: tổng 410 + 435 + 460 + 520 + 580 + 640 = 3.045; trung bình 3.045 ÷ 6 = 507,5; tháng cao nhất là tháng 6.
   Descriptive: total 410 + 435 + 460 + 520 + 580 + 640 = 3,045; average 3,045 ÷ 6 = 507.5; the highest month is June.
2. Chẩn đoán: tháng 6 hơn tháng 5 là 640 − 580 = 60. Mã QR tăng 410 − 340 = 70, ứng dụng giảm 230 − 240 = −10. Phần tăng đến từ mã QR.
   Diagnostic: June exceeds May by 640 − 580 = 60. QR code rose 410 − 340 = 70 and the app fell 230 − 240 = −10. The rise came from QR code.
3. Dự báo: ba mức tăng gần nhất đều là 60, trung bình 60. Tháng 7 = 640 + 60 = 700. Công thức (1.3) viết điều này.
   Predictive: the last three rises are all 60, average 60. July = 640 + 60 = 700. Formula (1.3) states this.

$$\hat{y}_{t+1} = y_t + \Delta_{tb} \qquad (1.3)$$

4. Đề xuất: ngưỡng là 90% × 750 = 675. Dự báo 700 vượt 675, nên mở rộng công suất trước tháng 7.
   Prescriptive: the threshold is 90% × 750 = 675. The forecast of 700 passes 675, so expand capacity before July.

```
TH = [410, 435, 460, 520, 580, 640]   # nghin giao dich moi thang
QR = {5: 340, 6: 410}
APP = {5: 240, 6: 230}
tong, tb = sum(TH), sum(TH) / len(TH)
print("Mo ta: tong", tong, "- trung binh", tb)
print("Mo ta: thang cao nhat", TH.index(max(TH)) + 1)
tang = TH[5] - TH[4]
print("Chan doan: tang", tang, "QR", QR[6] - QR[5],
      "app", APP[6] - APP[5])
muc = [TH[i] - TH[i - 1] for i in range(3, 6)]
du_bao = TH[5] + sum(muc) / 3
print("Du bao thang 7:", du_bao)
nguong = 0.9 * 750
print("De xuat:", "mo rong" if du_bao > nguong else "giu nguyen")
```

Kết quả chạy / *Output*

```
Mo ta: tong 3045 - trung binh 507.5
Mo ta: thang cao nhat 6
Chan doan: tang 60 QR 70 app -10
Du bao thang 7: 700.0
De xuat: mo rong
```

Máy cho cùng kết quả với tay. Dự báo 700 chỉ đúng nếu mức tăng không đổi, nên câu kết luận gửi chị Thảo ghi rõ giả định ấy.

The machine matches the hand work. The forecast of 700 holds only if the rise stays the same, so the conclusion sent to Ms Thảo states that assumption.

> **LỖI HAY GẶP / COMMON MISTAKE –** Nhảy thẳng tới dự báo mà bỏ qua mô tả và chẩn đoán. Nếu chưa biết tăng đến từ kênh nào, dự báo dễ lệch, và đề xuất mở rộng sai chỗ sẽ tốn tiền.
> *Jumping straight to prediction and skipping description and diagnosis. If you do not know which channel drove the rise, the forecast drifts, and an expansion in the wrong place wastes money.*

1.9. Việc làm và kỹ năng – *1.9. Jobs and skills*

Việc dùng AI và dữ liệu lớn đòi những kỹ năng mà chỉ một nhóm nhỏ người làm tài chính hiện có. Doanh nghiệp cần người vừa biết nghiệp vụ tài chính vừa biết lập trình và thống kê. Vai trò tách rời, một bên chuyên công nghệ thông tin và một bên chuyên tài chính, sẽ nhường chỗ cho người nắm cả hai. Cán bộ tuân thủ và quản lý rủi ro cũng phải hiểu mô hình hoạt động ra sao để kiểm tra, chất vấn và duyệt cho dùng. Lãnh đạo chịu trách nhiệm cuối cùng nên phải theo dõi được quá trình dựng và triển khai.

Using AI and big data demands skills that only a small group of finance practitioners hold today. Firms need people who know finance and also programming and statistics. The separate roles, one side expert in information technology and the other in finance, will give way to people who command both. Compliance staff and risk managers must also understand how a model works so they can audit, challenge and approve its use. Leaders who carry final responsibility must be able to follow how it is built and deployed.

Về việc làm có hai phía cùng tồn tại. Một phía là nhu cầu lớn về người giỏi phương pháp AI, toán, kỹ thuật phần mềm và khoa học dữ liệu. Phía kia là khả năng một số công việc lặp lại bị máy thay, nhiều lãnh đạo ngành tài chính dự đoán điều này. Trong vài năm tới, mô hình AI còn chạy song song với mô hình truyền thống, nên người làm nghề có thời

gian tích lũy kinh nghiệm. Cơ quan quản lý cũng cần nâng kỹ năng để kiểm tra các hệ thống AI.

Employment has two sides that coexist. One is strong demand for people skilled in AI methods, mathematics, software engineering and data science. The other is the chance that some repetitive jobs will be replaced by machines, which many finance leaders expect. For the next few years AI models will run alongside traditional ones, so practitioners have time to build experience. Regulators also need to raise their skills to inspect AI systems.

AI trong tài chính nên là công cụ bổ trợ cho người. Máy gợi ý, người quyết định, nhất là ở việc lớn như cho vay. Khi hệ thống tự động ngừng chạy, phải có người được đào tạo sẵn để đỡ chỗ, vì thị trường không được gián đoạn. Bảng sau mô tả bốn vị trí mà Bảo Ngọc có thể hướng tới.

AI in finance should be a tool that supports people. The machine suggests and the person decides, above all on big matters such as lending. When an automated system stops, trained people must be ready to step in, because markets cannot be left without service. The next table describes four positions Bảo Ngọc could aim for.

Vị trí *Position*	**Việc hằng ngày** *Daily work*	**Kỹ năng cần có** *Skills needed*
Chuyên viên quản lý dữ liệu tài chính *Financial data officer*	Theo dõi chất lượng bảng, ghi từ điển dữ liệu, xử lý yêu cầu truy cập *Watch table quality, keep the data dictionary, handle access requests*	SQL, hiểu nghiệp vụ, tỉ mỉ, biết quy định bảo vệ dữ liệu *SQL, business sense, care, knowledge of data protection rules*
Chuyên gia phân tích dữ liệu tài chính *Financial data analyst*	Nhận câu hỏi từ nghiệp vụ, tính chỉ số, dựng báo cáo, giải thích kết quả *Take questions from the business, compute metrics, build reports, explain results*	Thống kê, Python, trình bày số liệu *Statistics, Python, presenting figures*
Kỹ sư dữ liệu lớn *Big data engineer*	Dựng đường ống dữ liệu, tối ưu lưu trữ và xử lý phân tán, theo dõi tiến trình chạy *Build data pipelines, tune storage and distributed processing, monitor runs*	Python, SQL, hệ phân tán, hệ điều hành *Python, SQL, distributed systems, operating systems*
Phát triển ứng dụng *Application developer*	Viết dịch vụ đưa mô hình vào sản phẩm, kiểm thử, vận hành *Write services that put models into products, test, operate*	Lập trình, thiết kế dịch vụ, bảo mật *Programming, service design, security*

1.10. Ứng dụng ở tổ chức tài chính: phiếu một trang cho Trúc Xanh Pay – *1.10. Application in a financial organisation: a one-page brief for Trúc Xanh Pay*

Chị Thảo giao Bảo Ngọc một việc: nên cho ví Trúc Xanh Pay thêm chức năng gắn cờ giao dịch bất thường ngay khi phát sinh hay không? Bảo Ngọc không nhảy vào kỹ thuật. Bạn trả lời lần lượt sáu câu, mỗi câu dùng một khái niệm vừa học, và gom thành phiếu một trang.

Ms Thảo gives Bảo Ngọc a task: should the Trúc Xanh Pay wallet add a function that flags unusual transactions the moment they occur? Bảo Ngọc does not jump into technique. She answers six questions in turn, each using an idea just learnt, and gathers them into a one-page brief.

Câu hỏi *Question*	**Trả lời của Bảo Ngọc** *Bảo Ngọc's answer*
Quyết định cần ra là gì? *What decision is needed?*	Có bật gắn cờ ngay hay không *Whether to switch on immediate flagging*
Loại phân tích nào? *Which kind of analytics?*	Dự báo (giao dịch này có bất thường không) rồi đề xuất (chặn, hỏi lại hay cho qua) *Predictive (is this transaction unusual) then prescriptive (block, ask or pass)*
Nguồn dữ liệu nào? *Which data sources?*	Giao dịch ví (có cấu trúc), nhật ký ứng dụng (bán cấu trúc) *Wallet transactions (structured), app logs (semi-structured)*
Chữ V nào gây khó? *Which Vs cause difficulty?*	Tốc độ: phải trả lời trong vài trăm mili giây; khối lượng: 1.500.000 giao dịch mỗi ngày *Velocity: answer within a few hundred milliseconds; volume: 1,500,000 transactions a day*
Rủi ro gì? *What risks?*	Gắn cờ nhầm làm phiền khách; dữ liệu đăng nhập là dữ liệu cá nhân *Wrong flags annoy customers; login data is personal data*
Con người ở đâu? *Where are the people?*	Nhân viên xem các giao dịch bị gắn cờ và quyết định cuối cùng *Staff review flagged transactions and take the final decision*

Phiếu chưa đủ để chị Thảo quyết, vì còn thiếu một con số: ai sẽ xem các cờ? Bảo Ngọc ước lượng như sau.

The brief is not enough for Ms Thảo to decide, because one number is missing: who will review the flags? Bảo Ngọc estimates as follows.

1. Giả sử mô hình gắn cờ 0,7% giao dịch, tức 7 trên 1.000.
   Suppose the model flags 0.7% of transactions, which is 7 in 1,000.
2. Số cờ mỗi ngày: 1.500.000 × 0,007 = 10.500.
   Flags per day: 1,500,000 × 0.007 = 10,500.
3. Một nhân viên xem 60 cờ mỗi giờ, làm 7 giờ, được 60 × 7 = 420 cờ mỗi ngày.
   One staff member reviews 60 flags an hour for 7 hours, which is 60 × 7 = 420 flags a day.
4. Số nhân viên cần: 10.500 ÷ 420 = 25.
   Staff needed: 10,500 ÷ 420 = 25.

Cách đọc: 0,7% nghe nhỏ nhưng đòi 25 người xem mỗi ngày. Nếu tỷ lệ cờ sai giảm một nửa, số người cần cũng giảm một nửa. Bảo Ngọc ghi vào phiếu: nên bật chức năng, với điều kiện đo tỷ lệ cờ sai hằng tuần và có đủ người xem; chương 4 dạy cách đo.

Reading it: 0.7% sounds small but needs 25 people reviewing each day. If the share of false flags is halved, the number of staff halves too. Bảo Ngọc writes on the brief: switch the function on, on

condition that the false-flag rate is measured weekly and enough reviewers are in place; chapter 4 teaches how to measure it.

Ở thị trường chứng khoán, cách làm tương tự. Công ty Chứng khoán Trúc Xanh có ba nguồn: lệnh và khớp lệnh (có cấu trúc), giá theo phút và tick (tần suất cao), tin tức (phi cấu trúc). Mỗi nguồn trả lời một câu hỏi khác nhau, như lệnh nào đặt bất thường, giá di chuyển ra sao, tin nào ảnh hưởng. Phiếu một trang cho từng câu hỏi giúp lãnh đạo nhìn ra quyết định nào cần số liệu nào, và nguồn nào chưa đáng giữ.

In the stock market the method is the same. Trúc Xanh Securities has three sources: orders and matches (structured), prices by the minute and ticks (high frequency), and news (unstructured). Each answers a different question, such as which orders look odd, how prices move, and which news matters. A one-page brief per question lets leaders see which decision needs which figures, and which source is not worth keeping.

1.11. Thực hành bằng Python – *1.11. Practising with Python*

Các đoạn mã trong sách chỉ dùng thư viện chuẩn của Python để chạy được trên mọi máy. Khi làm thật, bạn sẽ dùng Apache Spark, pandas hoặc một cơ sở dữ liệu lớn; ý tưởng đọc từng phần mà bạn học ở đây vẫn y nguyên. Đoạn mã đầu sinh ra một tệp CSV 100.000 dòng giao dịch ví rồi đọc từng dòng bằng một hàm generator. Generator là hàm trả về từng phần tử một khi được hỏi, không dựng cả danh sách; nhờ từ khóa yield. Dòng đầu tệp là tên cột nên phải bỏ qua.

The code in this book uses only the Python standard library so it runs on any machine. In real work you will use Apache Spark, pandas or a large database; the idea of reading in pieces that you learn here stays the same. The first code creates a CSV file of 100,000 wallet transaction rows and reads it line by line with a generator function. A generator is a function that hands back one item at a time when asked, without building the whole list; the keyword yield does this. The first line of the file holds the column names, so it is skipped.

```
import csv, os, random, tempfile
random.seed(11)
KENH = ["QR", "app", "the", "quay"]
duong = os.path.join(tempfile.mkdtemp(), "vi.csv")
with open(duong, "w", newline="") as f:
    w = csv.writer(f)
    w.writerow(["ma", "so_tien", "kenh"])
    for i in range(100000):
        w.writerow([f"G{i:06d}", random.randint(10, 5000) * 1000,
                    random.choice(KENH)])
def doc_dong(path):
    with open(path, newline="") as f:
        r = csv.reader(f)
        next(r)
        for dong in r:
            yield dong
n, tong, lon, theo = 0, 0, 0, {}
for ma, tien, kenh in doc_dong(duong):
    tien = int(tien)
    n += 1
    tong += tien
    lon = max(lon, tien)
    theo[kenh] = theo.get(kenh, 0) + 1
print("so dong:", n)
print("tong tien:", tong)
```

```
print("lon nhat:", lon)
print("theo kenh:", theo)
```

Kết quả chạy / *Output*

```
so dong: 100000
tong tien: 249976631000
lon nhat: 5000000
theo kenh: {'quay': 25023, 'app': 24883, 'QR': 24891, 'the': 25203}
```

Tệp có 100.000 dòng, tổng 249.976.631.000 đồng, giao dịch lớn nhất 5.000.000 đồng. Số dòng theo kênh: mã QR 24.891, ứng dụng 24.883, thẻ 25.203, quầy 25.023. Vòng for chỉ giữ một dòng trong tay mỗi lần, và bốn biến đếm nhỏ giữ kết quả dồn. Cùng đoạn mã đó chạy được với tệp 100 triệu dòng, chỉ chậm hơn.

The file has 100,000 rows, a total of 249,976,631,000 dong, and a largest transaction of 5,000,000 dong. Rows by channel: QR code 24,891, app 24,883, card 25,203, counter 25,023. The for loop holds just one row at a time, and four small counters keep the running result. The same code would run on a file of 100 million rows, only more slowly.

Bây giờ đo lợi ích của cách đọc từng dòng. Đoạn mã thứ hai dùng tracemalloc, mô-đun chuẩn đo bộ nhớ Python đang dùng, để so hai cách: nạp cả tệp vào một danh sách rồi cộng, và cộng ngay khi đọc từng dòng.

Now measure the benefit of reading line by line. The second code uses tracemalloc, a standard module that measures the memory Python is using, to compare two ways: load the whole file into a list and then add, or add as each line is read.

```
import csv, os, random, tempfile, tracemalloc
random.seed(11)
duong = os.path.join(tempfile.mkdtemp(), "vi.csv")
with open(duong, "w", newline="") as f:
    w = csv.writer(f)
    for i in range(100000):
        w.writerow([f"G{i:06d}", random.randint(10, 5000) * 1000])
def doc_dong(path):
    with open(path, newline="") as f:
        for dong in csv.reader(f):
            yield dong
def nap_het():
    return sum(int(d[1]) for d in list(doc_dong(duong)))
def tung_dong():
    return sum(int(d[1]) for d in doc_dong(duong))
def do_bo_nho(ham):
    tracemalloc.start()
    kq = ham()
    dinh = tracemalloc.get_traced_memory()[1]
    tracemalloc.stop()
    return kq, dinh
k1, a = do_bo_nho(nap_het)
k2, b = do_bo_nho(tung_dong)
print("cung ket qua:", k1 == k2)
print(f"nap het: {a / 1e6:.1f} MB")
print(f"tung dong: {b / 1e3:.1f} KB")
print(f"ti le: {a / b:.0f}")
```

Kết quả chạy / *Output*

```
cung ket qua: True
nap het: 19.2 MB
tung dong: 47.5 KB
ti le: 404
```

Hai cách cho cùng tổng. Cách nạp hết dùng bộ nhớ đỉnh khoảng 19,2 MB, cách từng dòng chỉ khoảng 47,5 KB, tức nhỏ hơn khoảng 404 lần. Tệp càng lớn khoảng cách càng rộng: cách nạp hết tăng theo số dòng, cách từng dòng gần như đứng yên. Với tệp 50 GB, chỉ cách thứ hai còn chạy được trên máy một người.

Both ways give the same total. Loading everything peaks at about 19.2 MB of memory, while the line-by-line way needs only about 47.5 KB, around 404 times less. The bigger the file, the wider the gap: the load-everything way grows with the number of rows and the line-by-line way barely moves. For a 50 GB file only the second still runs on one person's machine.

Đoạn mã thứ ba làm việc của bước 4 ở mục 1.1: đếm dòng trùng mã, dòng thiếu số tiền và dòng số tiền âm trên 100.000 giao dịch ví có cài sẵn lỗi. Trước khi chạy, làm tay trên tám dòng. Mã giao dịch G03 xuất hiện hai lần, số tiền ở dòng 3 bị thiếu, số tiền dòng 6 là âm.

The third code does the work of step 4 in section 1.1: count rows with a repeated reference, a missing amount or a negative amount across 100,000 wallet transactions with planted faults. Before running it, work by hand on eight rows. Reference G03 appears twice, the amount in row 3 is missing and the amount in row 6 is negative.

Dòng *Row*	**Mã** *Reference*	**Số tiền (nghìn đồng)** *Amount (thousand dong)*	**Vấn đề** *Problem*
1	G01	50	Không *None*
2	G02	120	Không *None*
3	G03	(thiếu *missing)*	Thiếu số tiền *Missing amount*
4	G03	80	Trùng mã với dòng 3 *Same reference as row 3*
5	G05	80	Không *None*
6	G06	−30	Số tiền âm *Negative amount*
7	G07	200	Không *None*
8	G08	15	Không *None*

Ba dòng có vấn đề là 3, 4 và 6. Gọi m là số dòng có vấn đề, mỗi dòng tính một lần, và n là tổng số dòng. Tỷ lệ dòng có vấn đề là p = m ÷ n.

Three rows have a problem: 3, 4 and 6. Let m be the number of rows with a problem, each counted once, and n the total rows. The share of rows with a problem is p = m ÷ n.

$$p = m \div n \tag{1.4}$$

Với bảng tám dòng, p = 3 ÷ 8 = 0,375, tức 37,5%. Giờ đến đoạn mã. Một dòng có hai lỗi vẫn chỉ tính một lần vào m, nên mã dùng một biến sai cho mỗi dòng.

For the eight-row table, p = 3 ÷ 8 = 0.375, or 37.5%. Now the code. A row with two faults still counts once in m, so the code uses one flag per row.

```
import random
random.seed(7)
rows, truoc = [], None
for i in range(100000):
    ma = f"G{i:06d}"
    if truoc and random.random() < 0.008:
        ma = truoc                # dong trung ma voi dong truoc
    tien = random.randint(10, 5000) * 1000
    x = random.random()
    if x < 0.005:
        tien = None               # thieu so tien
    elif x < 0.007:
        tien = -tien              # so tien am
    rows.append((ma, tien))
    truoc = ma
thay, trung, thieu, am, loi = set(), 0, 0, 0, 0
for ma, tien in rows:
    sai = False
    if ma in thay:
        trung, sai = trung + 1, True
    thay.add(ma)
    if tien is None:
        thieu, sai = thieu + 1, True
    elif tien < 0:
        am, sai = am + 1, True
    loi += sai
print("so dong:", len(rows))
print("trung ma:", trung, "thieu tien:", thieu, "am:", am)
print("dong co van de:", loi)
print(f"ti le p: {loi / len(rows):.2%}")
```

Kết quả chạy / *Output*

```
so dong: 100000
trung ma: 782 thieu tien: 505 am: 186
dong co van de: 1469
ti le p: 1.47%
```

Trong 100.000 dòng có 782 dòng trùng mã, 505 dòng thiếu số tiền và 186 dòng số tiền âm. Tổng ba loại là 1.473 nhưng số dòng có vấn đề chỉ là 1.469, vì một số dòng dính nhiều lỗi cùng lúc. Tỷ lệ p là 1,47%. Nghĩa là cứ 1.000 dòng có khoảng 14,7 dòng cần xử lý trước khi đưa vào mô hình.

Among 100,000 rows, 782 repeat a reference, 505 lack an amount and 186 have a negative amount. The three counts add up to 1,473 but the rows with a problem number only 1,469, because some rows carry several faults at once. The share p is 1.47%. That means about 14.7 in every 1,000 rows need handling before they go into a model.

> **LỖI HAY GẶP / COMMON MISTAKE –** Nạp cả tệp lớn vào danh sách rồi mới tính. Máy chạy được với tệp thử 100.000 dòng nhưng hết bộ nhớ ở tệp thật. Hãy viết ngay từ đầu bằng generator, và luôn đếm số dòng đã đọc để so với số dòng mong đợi.
> *Loading a whole large file into a list before computing. The code runs on a 100,000-row test file but runs out of memory on the real one. Write it with a generator from the start, and always count the rows read against the number expected.*

PHẢI THUỘC / MEMORISE – Generator đọc từng dòng, bộ nhớ gần như không đổi theo kích thước tệp. Tỷ lệ dòng có vấn đề p = m ÷ n, mỗi dòng tính một lần dù dính nhiều lỗi. Kiểm chất lượng trước khi huấn luyện mô hình.
A generator reads line by line and memory barely changes with file size. The share of rows with a problem p = m ÷ n, each row counted once however many faults it has. Check quality before training a model.

Bảng thuật ngữ – *Glossary*

Thuật ngữ *Term (Vietnamese)*	**Tiếng Anh** *English*	**Giải thích ngắn** *Short definition*
Dữ liệu lớn	Big data	Dữ liệu nhiều, đến nhanh, nhiều dạng, vượt khả năng của công cụ thông thường. *Data so large, fast or varied that ordinary tools struggle.*
Khối lượng	Volume	Chữ V về lượng dữ liệu, đo bằng byte. *The V for how much data there is, measured in bytes.*
Tốc độ	Velocity	Chữ V về nhịp dữ liệu đến và thời gian được phép xử lý. *The V for how fast data arrives and how soon it must be handled.*
Đa dạng	Variety	Chữ V về nhiều dạng và nhiều nguồn dữ liệu. *The V for many forms and sources of data.*
Tính xác thực	Veracity	Chữ V về độ đáng tin của dữ liệu. *The V for how far data can be trusted.*
Giá trị	Value	Chữ V về ích lợi dữ liệu mang lại cho quyết định. *The V for the benefit data brings to decisions.*
Byte	Byte	Đơn vị nhỏ nhất thường dùng để đo dung lượng, đủ lưu một chữ cái. *The basic unit of storage, enough for one letter.*
Bản ghi	Record	Một dòng dữ liệu, ví dụ một giao dịch. *One row of data, such as one transaction.*
Dữ liệu có cấu trúc	Structured data	Dữ liệu xếp thành hàng và cột với kiểu cố định. *Data in rows and columns with fixed types.*
Dữ liệu bán cấu trúc	Semi-structured data	Dữ liệu có nhãn nhưng mỗi bản ghi có thể khác khóa. *Labelled data whose records may carry different keys.*
Dữ liệu phi cấu trúc	Unstructured data	Văn bản, ảnh, âm thanh không có khuôn bảng. *Text, images and audio with no table layout.*

Thuật ngữ *Term (Vietnamese)*	**Tiếng Anh** *English*	**Giải thích ngắn** *Short definition*
Dữ liệu thay thế	Alternative data	Dữ liệu ngoài hồ sơ tài chính truyền thống. *Data from outside traditional financial records.*
Trí tuệ nhân tạo (AI)	Artificial intelligence (AI)	Hệ thống máy đưa ra dự đoán, gợi ý hoặc quyết định cho mục tiêu do người đặt. *A machine system that predicts, recommends or decides for goals set by people.*
Học máy (ML)	Machine learning (ML)	Phần của AI tự rút quy luật từ dữ liệu, không cần người viết từng luật. *The part of AI that learns patterns from data without people writing each rule.*
Học sâu	Deep learning	Học máy dùng mạng nơ-ron nhiều tầng. *Machine learning with many-layered neural networks.*
Học có giám sát	Supervised learning	Học từ ví dụ đã có đáp án. *Learning from examples that carry answers.*
Học không giám sát	Unsupervised learning	Tìm nhóm hoặc quy luật trong dữ liệu chưa có nhãn. *Finding groups or patterns in unlabelled data.*
Học tăng cường	Reinforcement learning	Học qua thử, nhận thưởng hoặc phạt. *Learning by trial, with rewards and penalties.*
Bốn loại phân tích	Four kinds of analytics	Mô tả (chuyện gì đã xảy ra), chẩn đoán (vì sao), dự báo (sẽ ra sao), đề xuất (nên làm gì). *Descriptive (what happened), diagnostic (why), predictive (what next), prescriptive (what to do).*
Điện toán đám mây	Cloud computing	Thuê máy chủ và kho lưu trữ qua mạng. *Renting servers and storage over a network.*
Con người trong vòng lặp	Human in the loop	Máy gợi ý, người duyệt và quyết định. *The machine suggests; a person reviews and decides.*

1.12. Bài tập vận dụng có lời giải – *Applied exercises with solutions*

Bài 1. Ước dung lượng lệnh chứng khoán – *Exercise 1. Estimating order storage*

Công ty Chứng khoán Trúc Xanh nhận 600.000 lệnh mỗi ngày, mỗi lệnh ghi 180 byte, một năm có 250 phiên. Hệ thống lưu mỗi dữ liệu thành 3 bản. Tính dung lượng một năm trước và sau khi nhân bản (1 GB = 1.000 MB). Nếu mỗi năm lượng lệnh không đổi, mấy năm thì vượt 1 TB?

Trúc Xanh Securities receives 600,000 orders a day, each record is 180 bytes, and a year has 250 trading sessions. The system stores every item in 3 copies. Compute one year of storage

before and after replication (1 GB = 1,000 MB). If the daily order count stays the same, after how many years does the total pass 1 TB?

1. Một ngày: 600.000 × 180 = 108.000.000 byte, tức 108 MB.
   One day: 600,000 × 180 = 108,000,000 bytes, which is 108 MB.
2. Một năm: 108 MB × 250 = 27.000 MB, tức 27 GB.
   One year: 108 MB × 250 = 27,000 MB, which is 27 GB.
3. Nhân bản 3 lần: 27 × 3 = 81 GB mỗi năm.
   Three copies: 27 × 3 = 81 GB per year.
4. 1 TB = 1.000 GB. Số năm = 1.000 ÷ 81 ≈ 12,35.
   1 TB = 1,000 GB. Years = 1,000 ÷ 81 ≈ 12.35.
5. Sau 12 năm có 972 GB, chưa tới 1 TB. Sang năm thứ 13 tổng là 1.053 GB, vượt 1 TB.
   After 12 years there are 972 GB, still under 1 TB. In year 13 the total is 1,053 GB, which passes 1 TB.

Bài 2. Kiểm tra chất lượng bảng 8 dòng – ***Exercise 2. Quality check on an 8-row table***

Bảng giao dịch có 8 dòng, ghi theo thứ tự mã và số tiền (nghìn đồng): (B1, 40); (B2, thiếu); (B3, 25); (B3, 25); (B5, 60); (B6, −5); (B6, −5); (B8, 90). Đếm dòng thiếu số tiền, dòng trùng mã (từ lần xuất hiện thứ hai), dòng số tiền âm. Tính tỷ lệ dòng có vấn đề và tổng số tiền của các dòng sạch.

A transaction table has 8 rows, listed as reference and amount (thousand dong): (B1, 40); (B2, missing); (B3, 25); (B3, 25); (B5, 60); (B6, −5); (B6, −5); (B8, 90). Count rows with a missing amount, rows with a repeated reference (from the second appearance) and rows with a negative amount. Compute the share of rows with a problem and the total amount of the clean rows.

1. Thiếu số tiền: dòng 2, tức 1 dòng.
   Missing amount: row 2, so 1 row.
2. Trùng mã: B3 lặp ở dòng 4, B6 lặp ở dòng 7, tức 2 dòng.
   Repeated reference: B3 repeats in row 4 and B6 in row 7, so 2 rows.
3. Số tiền âm: dòng 6 và dòng 7, tức 2 dòng.
   Negative amount: rows 6 and 7, so 2 rows.
4. Dòng có vấn đề là 2, 4, 6, 7 (dòng 7 bị tính một lần dù dính hai lỗi): 4 trên 8, tức 50%.
   Rows with a problem are 2, 4, 6 and 7 (row 7 counts once although it has two faults): 4 of 8, or 50%.
5. Dòng sạch là 1, 3, 5, 8. Tổng = 40 + 25 + 60 + 90 = 215 nghìn đồng.
   Clean rows are 1, 3, 5 and 8. Total = 40 + 25 + 60 + 90 = 215 thousand dong.

Bài 3. Bốn loại phân tích cho cụm máy rút tiền – ***Exercise 3. Four kinds of analytics for a cash machine cluster***

Số lượt rút tiền tại một cụm máy rút tiền tự động (ATM, automated teller machine) của Ngân hàng Trúc Xanh trong 5 tuần liên tiếp (nghìn lượt): 120; 126; 132; 138; 144. Mỗi máy phục vụ tối đa 40 nghìn lượt mỗi tuần. Tính trung bình 5 tuần (mô tả), mức tăng mỗi tuần (chẩn đoán), dự báo tuần 6 nếu mức tăng giữ nguyên, rồi số máy cần có (đề xuất).

Cash withdrawals at one cluster of automated teller machines (ATMs) of Trúc Xanh Bank over 5 consecutive weeks (thousand withdrawals): 120; 126; 132; 138; 144. Each machine serves at most 40 thousand withdrawals a week. Compute the 5-week average (descriptive), the weekly rise (diagnostic), the week-6 forecast if the rise holds (predictive), then the number of machines needed (prescriptive).

1. Mô tả: (120 + 126 + 132 + 138 + 144) ÷ 5 = 660 ÷ 5 = 132 nghìn lượt.
   Descriptive: (120 + 126 + 132 + 138 + 144) ÷ 5 = 660 ÷ 5 = 132 thousand.
2. Chẩn đoán: hiệu hai tuần liền kề đều là 6, nên lượt rút tăng đều 6 nghìn mỗi tuần.
   Diagnostic: every pair of adjacent weeks differs by 6, so withdrawals rise by 6 thousand a week.
3. Dự báo: tuần 6 = 144 + 6 = 150 nghìn lượt.
   Predictive: week 6 = 144 + 6 = 150 thousand.
4. Đề xuất: 150 ÷ 40 = 3,75, làm tròn lên 4 máy.
   Prescriptive: 150 ÷ 40 = 3.75, rounded up to 4 machines.
5. Giới hạn: dự báo giả định mức tăng không đổi. Nếu có ngày lễ, cần xem lại.
   Limit: the forecast assumes a constant rise. If a holiday falls in the week, revisit it.

1.13. Câu hỏi tự luận và dàn ý trả lời – *Essay questions and answer outlines*

Câu 1. Trình bày năm chữ V của dữ liệu lớn. Cho mỗi chữ một ví dụ ở ngân hàng hoặc công ty chứng khoán.

Question 1. *Describe the five Vs of big data. Give each one an example from a bank or a securities company.*

- Khối lượng: dữ liệu nhiều đến mức một máy không chứa nổi; ví dụ nhật ký ứng dụng của ngân hàng số.
  Volume: so much data that one machine cannot hold it; for example a digital bank's app logs.
- Tốc độ: dữ liệu đến liên tục và phải xử lý kịp; ví dụ giá khớp lệnh trong phiên.
  Velocity: data arrives continuously and must be handled in time; for example trade prices during the session.
- Đa dạng: nhiều dạng dữ liệu khác nhau; ví dụ bảng giao dịch, tin tức, ảnh giấy tờ.
  Variety: many forms of data; for example transaction tables, news and photos of documents.
- Tính xác thực: dữ liệu đáng tin đến đâu; ví dụ tài khoản tạo bằng thông tin giả.
  Veracity: how far the data can be trusted; for example accounts opened with false details.
- Giá trị: dữ liệu phải dẫn tới quyết định có ích; ví dụ giảm giao dịch gian lận.
  Value: data must lead to a useful decision; for example fewer fraudulent transactions.

Câu 2. So sánh dữ liệu có cấu trúc, bán cấu trúc và phi cấu trúc. Nêu nguồn của từng loại ở Ngân hàng Trúc Xanh và một khó khăn khi xử lý dữ liệu phi cấu trúc.

Question 2. *Compare structured, semi-structured and unstructured data. Name a source of each at Trúc Xanh Bank and one difficulty with unstructured data.*

- Có cấu trúc: cột và kiểu cố định; ví dụ bảng giao dịch trong hệ thống lõi.
  Structured: fixed columns and types; for example the transaction table in the core system.

- Bán cấu trúc: có nhãn nhưng mỗi bản ghi có thể khác khóa; ví dụ nhật ký JSON của ứng dụng.
  Semi-structured: labelled, but each record may carry different keys; for example JSON app logs.
- Phi cấu trúc: không có khuôn sẵn; ví dụ email khiếu nại, ảnh giấy tờ, ghi âm tổng đài.
  Unstructured: no ready template; for example complaint emails, document photos and call recordings.
- Khó khăn: phải biến văn bản, ảnh, giọng nói thành con số trước khi tính.
  Difficulty: text, images and speech must be turned into numbers before any calculation.
- Cách xử lý chung: chuyển dần sang dạng có cấu trúc rồi mới tổng hợp.
  Common approach: convert step by step to a structured form, then summarise.

Câu 3. Phân tích quan hệ giữa AI, học máy, học sâu và dữ liệu lớn. Nêu ba lợi ích và ba rủi ro khi dùng chúng trong tài chính, và giải thích vì sao vẫn cần con người trong vòng lặp.

Question 3. *Explain how AI, machine learning, deep learning and big data relate. Give three benefits and three risks of using them in finance, and say why people must stay in the loop.*

- Quan hệ: học sâu nằm trong học máy, học máy nằm trong AI; dữ liệu lớn là nhiên liệu để huấn luyện.
  Relation: deep learning sits inside machine learning, which sits inside AI; big data is the fuel for training.
- Lợi ích: giảm chi phí xử lý, thẩm định nhanh hơn, mở dịch vụ cho người chưa có lịch sử tín dụng.
  Benefits: lower processing cost, faster assessment, services for people with no credit history.
- Rủi ro: dữ liệu kém chất lượng gây thiên lệch, mô hình khó giải thích, nhiều bên cùng dùng một mô hình.
  Risks: poor data causes bias, models are hard to explain, many firms use the same model.
- Rủi ro khác: dữ liệu cá nhân bị lộ, phụ thuộc vài nhà cung cấp, thiếu người đủ kỹ năng.
  Other risks: personal data exposure, dependence on a few providers, too few skilled people.
- Con người trong vòng lặp: máy gợi ý, người quyết định ở việc lớn như cho vay.
  People in the loop: the machine suggests and a person decides on big matters such as lending.
- Người còn phải thay máy khi hệ thống ngừng chạy.
  People must also step in when the system stops.

1.14. Câu hỏi trắc nghiệm – *Multiple-choice questions*

1. Một tệp có 2.500.000 KB (1 KB = 1.000 byte). Tệp đó bằng bao nhiêu GB?
A file has 2,500,000 KB (1 KB = 1,000 bytes). How many GB is that?

A. 25 GB
B. 2,5 GB / *2.5 GB*
C. 0,25 GB / *0.25 GB*
D. 250 GB

2. Nhiều tài khoản bị mở bằng thông tin giả nên số liệu khách hàng khó tin. Chữ V nào bị ảnh hưởng trực tiếp nhất?
Many accounts were opened with false details, so customer figures are hard to trust. Which V is hit most directly?

A. Khối lượng / *Volume*
B. Tốc độ / *Velocity*
C. Tính xác thực / *Veracity*
D. Đa dạng / *Variety*

3. Ví có 2.000.000 giao dịch mỗi ngày, mỗi bản ghi 150 byte. Dung lượng một năm 365 ngày là bao nhiêu?
A wallet has 2,000,000 transactions a day, each record 150 bytes. How much storage does a 365-day year need?

A. 109,5 GB / *109.5 GB*
B. 10,95 GB / *10.95 GB*
C. 1,095 TB / *1.095 TB*
D. 109,5 TB / *109.5 TB*

4. Mỗi dòng nhật ký ứng dụng ghi dạng JSON, có dòng thêm khóa ghi chú, có dòng không. Đó là dữ liệu loại nào?
Each app log line is JSON; some lines add a note key and some do not. What kind of data is this?

A. Có cấu trúc / *Structured*
B. Phi cấu trúc / *Unstructured*
C. Không thuộc loại nào / *None of these*
D. Bán cấu trúc / *Semi-structured*

5. Trúc Xanh muốn tự chia khách thành các nhóm giống nhau về thói quen chi tiêu và chưa có nhãn nào. Nên dùng cách học nào?
Trúc Xanh wants to group customers by similar spending habits, with no labels yet. Which kind of learning fits?

A. Học có giám sát / *Supervised learning*
B. Học tăng cường / *Reinforcement learning*
C. Hồi quy tuyến tính / *Linear regression*
D. Học không giám sát / *Unsupervised learning*

6. Số giao dịch ba tháng liền là 300, 330 và 360 nghìn, mỗi tháng tăng thêm cùng mức. Dự báo tháng thứ tư là bao nhiêu nghìn?
Transactions over three months are 300, 330 and 360 thousand, rising by the same amount each month. What is the month-4 forecast in thousands?

A. 390
B. 400
C. 380
D. 420

7. Câu hỏi "tháng sau nên tăng hạn mức cho nhóm khách nào" thuộc loại phân tích nào?
The question "which customer group should get a higher limit next month" belongs to which kind of analytics?

A. Mô tả / *Descriptive*
B. Chẩn đoán / *Diagnostic*
C. Đề xuất / *Prescriptive*
D. Dự báo / *Predictive*

8. Bảng 8 dòng có 1 dòng thiếu số tiền, 1 dòng trùng mã và 1 dòng số tiền âm, ba dòng khác nhau. Tỷ lệ dòng cần xử lý là bao nhiêu?
An 8-row table has 1 row with a missing amount, 1 with a repeated reference and 1 with a negative amount, all different rows. What share of rows needs fixing?

A. 25%
B. 37,5% / *37.5%*
C. 50%
D. 12,5% / *12.5%*

Chương 2. Hạ tầng và công cụ xử lý dữ liệu lớn – *Chapter 2. Big data infrastructure and tools*

Mục tiêu chương – ***Chapter objectives***

- Phân biệt mở rộng dọc và mở rộng ngang, ước lượng thời gian quét dữ liệu trên cụm máy (nhiều máy nối mạng làm chung một việc).
  Tell vertical from horizontal scaling and estimate the scanning time on a cluster (many networked machines sharing one job).
- Tính số khối và dung lượng thực chiếm trên HDFS (hệ thống tệp lưu mỗi khối ở nhiều máy), giải thích vì sao một máy hỏng không làm mất dữ liệu.
  Compute blocks and raw storage on HDFS (a file system that keeps each block on many machines) and explain why one failed machine loses no data.
- Tính tay và mô phỏng MapReduce (đếm và gộp bằng nhiều máy làm song song), bộ kết hợp (gộp trước khi gửi), phân vùng (chia bảng thành phần), nối phát tán (gửi bản sao bảng nhỏ đến mọi máy); đo và giảm lệch dữ liệu.
  Work by hand and simulate MapReduce (counting and merging with many machines in parallel), combiners (merging before sending), partitions (cutting a table into parts) and broadcast joins (sending a copy of the small table to every machine); measure and reduce data skew.
- Đọc kế hoạch Spark (công cụ tính trong bộ nhớ của cụm), chọn loại NoSQL (cơ sở dữ liệu phi quan hệ), định dạng lưu và kiểu kho dữ liệu theo bài toán.
  Read a Spark plan (Spark computes in the cluster's memory) and choose a NoSQL type (a non-relational database), a storage format and a data store style for a problem.
- Nêu rủi ro khi thuê hạ tầng đám mây, chia trách nhiệm giữa hai bên và cách giảm rủi ro.
  State the risks of renting cloud infrastructure, how responsibility is shared and how to reduce the risk.

Bảo Ngọc thực tập ở Trung tâm Phân tích dữ liệu của Ngân hàng TMCP Trúc Xanh. Sáng thứ Hai, anh Quân dẫn bạn xuống phòng máy và nói: báo cáo giao dịch năm ngoái chạy gần ba tiếng trên một máy, năm nay dữ liệu gấp đôi, em tính xem làm thế nào. Chị Thảo, giám đốc khối Quản trị rủi ro, dặn thêm: số nào lên báo cáo cũng phải chỉ ra được nó lấy từ đâu. Bạn đi cùng Bảo Ngọc qua từng lớp của hạ tầng: cụm máy, nơi lưu dữ liệu, cách chia việc, công cụ tính, và nơi đặt hệ thống.

Bảo Ngọc is an intern at the Data Analytics Centre of Trúc Xanh Joint Stock Commercial Bank. On Monday morning Mr Quân takes her to the machine room and says that last year's transaction report ran for nearly three hours on one machine, this year the data has doubled, and she should work out what to do. Ms Thảo, director of Risk Management, adds that every figure in a report must be traceable to its source. You follow Bảo Ngọc through each layer of the infrastructure: the cluster, where the data is stored, how work is divided, the computing tools, and where the system runs.

2.1. Mở rộng dọc, mở rộng ngang và cụm máy – *2.1. Scaling up, scaling out and clusters*

Quán phở đông khách có hai cách phục vụ kịp. Cách một là mua nồi nước dùng to hơn và bếp mạnh hơn. Cách hai là mở thêm nhiều bếp nhỏ đặt cạnh nhau. Máy tính cũng có hai cách như vậy.

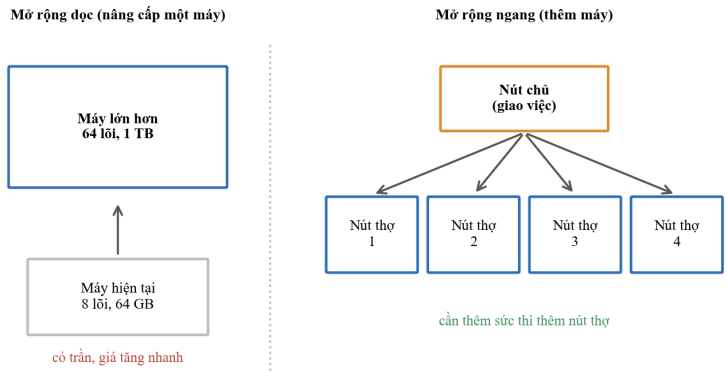
A busy noodle shop has two ways to keep up. One is to buy a bigger stock pot and a stronger stove. The other is to open several small kitchens side by side. Computers have the same two options.

Mở rộng dọc (vertical scaling, scale up) là nâng cấp một máy: thêm lõi xử lý (CPU, bộ phận tính toán), thêm bộ nhớ trong (RAM, nơi dữ liệu nằm tạm khi máy đang tính), thêm ổ đĩa. Cách này dễ vì chương trình giữ nguyên. Nhưng máy có trần, máy càng lớn giá càng tăng nhanh, và máy hỏng thì mọi việc dừng.

Vertical scaling (scale up) means upgrading one machine: more processor cores (CPU, the part that calculates), more memory (RAM, where data sits temporarily while the machine works) and more disk. It is easy because the programs stay the same. But a machine has a ceiling, the price climbs steeply as it grows, and when it fails all work stops.

Mở rộng ngang (horizontal scaling, scale out) là thêm nhiều máy thường, mỗi máy làm một phần việc. Nhiều máy nối mạng làm việc như một hệ gọi là cụm máy (cluster). Mỗi máy trong cụm là một nút (node). Thường có một nút chủ (master) giao việc và nhiều nút thợ (worker) làm việc. Một nút hỏng thì cụm vẫn chạy, nhưng chương trình phải biết chia việc nên khó viết hơn.

Horizontal scaling (scale out) means adding many ordinary machines, each doing part of the work. Many networked machines that behave as one system form a cluster. Each machine in the cluster is a node. Usually one master node hands out the work and many worker nodes do it. If a node fails the cluster keeps running, but the program must know how to split the work, so it is harder to write.



Hình 2.1. Bên trái, một máy được thay bằng máy lớn hơn. Bên phải, một nút chủ giao việc cho bốn nút thợ; muốn thêm sức thì thêm nút thợ.

Figure 2.1. *On the left, one machine is replaced by a bigger one. On the right, one master node hands work to four worker nodes; to gain power, add workers.*

Tiêu chí *Criterion*	**Mở rộng dọc** *Scale up*	**Mở rộng ngang** *Scale out*
Cách làm *How*	Thay bằng máy lớn hơn *Replace with a bigger machine*	Thêm nút vào cụm *Add nodes to the cluster*
Trần *Ceiling*	Có: máy lớn nhất đang bán *Yes: the biggest machine on sale*	Gần như không: thêm nút tiếp *Almost none: keep adding nodes*
Một máy hỏng *One machine fails*	Cả hệ dừng *The whole system stops*	Cụm chạy tiếp, việc được giao lại *The cluster carries on and work is reassigned*
Chi phí khi lớn *Cost at scale*	Tăng nhanh *Rises steeply*	Tăng gần tỷ lệ với số nút *Rises roughly with the node count*
Công sức lập trình *Programming effort*	Ít *Low*	Nhiều: phải chia việc và gom kết quả *High: work must be split and results merged*
Hợp với *Fits*	Dữ liệu vừa trong một máy *Data that fits in one machine*	Dữ liệu vượt quá một máy *Data larger than one machine*

Bài mẫu 1: thời gian quét dữ liệu – ***Worked example 1: scanning time***

Dữ liệu lệnh khớp một năm của Công ty Chứng khoán Trúc Xanh là 2 TB, tức 2.000.000 MB. Mỗi máy đọc đĩa 200 MB mỗi giây. Hỏi một máy và 20 máy đọc hết dữ liệu trong bao lâu?

A year of order-matching data at Trúc Xanh Securities is 2 TB, or 2,000,000 MB. Each machine reads its disk at 200 MB per second. How long do one machine and 20 machines take to read it all?

1. Một máy: 2.000.000 ÷ 200 = 10.000 giây, khoảng 2 giờ 47 phút.
   One machine: 2,000,000 ÷ 200 = 10,000 seconds, about 2 hours 47 minutes.
2. Với 20 máy, mỗi máy giữ 2.000.000 ÷ 20 = 100.000 MB.
   With 20 machines, each holds 2,000,000 ÷ 20 = 100,000 MB.
3. Mỗi máy đọc 100.000 ÷ 200 = 500 giây, tức 8 phút 20 giây. Hai mươi máy đọc cùng lúc.
   Each machine reads 100,000 ÷ 200 = 500 seconds, or 8 minutes 20 seconds. All 20 read at the same time.
4. Cụm còn phải chia việc và gom kết quả. Giả sử mỗi nút tốn thêm 2 giây điều phối: 20 ∗ 2 = 40 giây. Tổng cộng 540 giây, tức 9 phút.
   The cluster also has to split the work and gather results. Assume each node costs 2 extra seconds of coordination: 20 ∗ 2 = 40 seconds. The total is 540 seconds, or 9 minutes.

$$t(n) = D \div (n * v) + c * n \tag{2.1}$$

Trong công thức (2.1), D là dung lượng cần đọc, v là tốc độ đọc của một nút, c là số giây điều phối mỗi nút thêm vào. Đoạn mã sau tính t(n) cho vài cỡ cụm và dò số nút cho thời gian ngắn nhất.

In formula (2.1), D is the data to read, v is the read speed of one node and c is the coordination seconds each extra node adds. This code computes t(n) for several cluster sizes and searches for the node count with the shortest time.

```
# Thời gian quét dữ liệu khi thêm nút
DU_LIEU = 2_000_000        # MB (2 TB)
TOC_DO = 200               # MB/giây mỗi nút
DIEU_PHOI = 2              # giây điều phối cho mỗi nút
def thoi_gian(n):
    return DU_LIEU / (n * TOC_DO) + DIEU_PHOI * n
for n in (1, 5, 20, 50, 100, 200):
    print(f"nút={n:>3}  giây={thoi_gian(n):>8.1f}")
tot = min(range(1, 301), key=thoi_gian)
print(f"nút tốt nhất: {tot}  giây={thoi_gian(tot):.1f}")
```

Kết quả chạy / *Output*

```
nút=  1  giây= 10002.0
nút=  5  giây=  2010.0
nút= 20  giây=   540.0
nút= 50  giây=   300.0
nút=100  giây=   300.0
nút=200  giây=   450.0
nút tốt nhất: 71  giây=282.8
```

Từ 1 lên 5 nút, thời gian giảm từ 10.002,0 xuống 2.010,0 giây. Từ 100 lên 200 nút, thời gian lại tăng từ 300,0 lên 450,0 giây vì phần điều phối tăng theo số nút. Số nút tốt nhất là 71, mất 282,8 giây. Thêm máy chỉ có lợi đến một mức.

From 1 to 5 nodes the time falls from 10,002.0 to 2,010.0 seconds. From 100 to 200 nodes it rises again from 300.0 to 450.0 seconds because coordination grows with the node count. The best size is 71 nodes at 282.8 seconds. Adding machines pays off only up to a point.

Chị Thảo hỏi Bảo Ngọc: thêm 20 máy thì có nhanh gấp 20 lần không? Bảo Ngọc trả lời bằng số. Từ 10.002 xuống 540 giây là nhanh khoảng 18,5 lần, thấp hơn 20 lần vì có 40 giây điều phối. Chị Thảo ghi mức tăng tốc kèm giả định điều phối 2 giây mỗi nút vào biên bản.

Ms Thảo asks Bảo Ngọc whether adding 20 machines makes the job 20 times faster. Bảo Ngọc answers with numbers. Going from 10,002 to 540 seconds is about 18.5 times faster, below 20 because of the 40 seconds of coordination. Ms Thảo records the speed-up together with the assumption of 2 seconds of coordination per node.

2.2. Hệ thống tệp phân tán HDFS – *2.2. The distributed file system HDFS*

Một thư viện lớn chia quyển sách dày thành nhiều tập. Mỗi tập được photo ba bản và đặt ở ba chi nhánh khác nhau. Một thủ thư giữ sổ ghi tập nào nằm ở đâu. HDFS (Hadoop Distributed File System, hệ thống tệp phân tán của Hadoop, một bộ công cụ mã mở cho dữ liệu lớn) làm đúng như vậy với tệp dữ liệu.

A large library splits a thick book into volumes. Each volume is photocopied three times and kept in three different branches. One librarian holds the register of which volume is where. HDFS (the Hadoop Distributed File System, part of Hadoop, an open-source toolkit for big data) does exactly this with data files.

HDFS cắt mỗi tệp thành các khối (block) cố định, mặc định 128 MB, và lưu mỗi khối ở nhiều máy. Việc lưu nhiều bản gọi là nhân bản (replication); mặc định mỗi khối có 3 bản.

Khối lớn giúp máy đọc liền một mạch thay vì nhảy qua nhiều mảnh nhỏ, và giữ sổ danh bạ gọn.

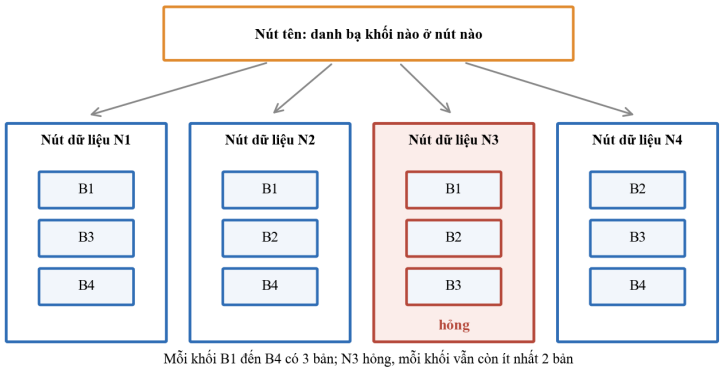
HDFS cuts every file into fixed blocks, 128 MB by default, and stores each block on several machines. Keeping several copies is called replication; by default every block has 3 copies. Large blocks let a machine read in one sweep instead of jumping across many small pieces, and keep the directory compact.

Có hai loại nút. Nút tên (NameNode) giữ danh bạ: tệp nào gồm khối nào, khối nào nằm ở nút nào. Nút này không giữ dữ liệu thật. Nút dữ liệu (DataNode) giữ các khối thật và cứ vài giây báo một tín hiệu còn sống, gọi là nhịp tim (heartbeat).

There are two kinds of node. The NameNode holds the directory: which file consists of which blocks and which node holds each block. It holds no real data. A DataNode holds the real blocks and every few seconds sends a signal that it is alive, called a heartbeat.

Đọc tệp: chương trình hỏi nút tên, nhận danh sách khối, rồi đọc thẳng từ nút dữ liệu gần nhất. Chịu lỗi (fault tolerance) hoạt động như sau: khi một nút dữ liệu mất nhịp tim khoảng 10 phút, nút tên coi nút đó đã chết và ra lệnh chép lại các khối thiếu từ những bản còn sống sang nút khác. Người dùng không phải làm gì.

Reading a file: the program asks the NameNode, receives the list of blocks, then reads straight from the nearest DataNode. Fault tolerance works like this: when a DataNode misses heartbeats for about 10 minutes, the NameNode treats it as dead and orders the missing blocks to be copied from surviving replicas to other nodes. The user does nothing.



Hình 2.2. Nút tên (cam) giữ danh bạ; bốn nút dữ liệu giữ các khối B1 đến B4, mỗi khối có 3 bản. Nút N3 hỏng (viền đỏ) nhưng mỗi khối vẫn còn bản ở các nút còn lại.

Figure 2.2. *The NameNode (orange) holds the directory; four DataNodes hold blocks B1 to B4, each in 3 copies. Node N3 has failed (red outline) but every block still has copies on the surviving nodes.*

Để tính dung lượng, bạn cần ba số: dung lượng tệp, kích thước khối và số bản sao.

To work out storage you need three numbers: the file size, the block size and the replication factor.

1. Chia dung lượng tệp cho kích thước khối rồi làm tròn lên: đó là số khối.
   Divide the file size by the block size and round up: that is the number of blocks.
2. Mọi khối đầy 128 MB trừ khối cuối, khối cuối bằng phần dư. HDFS không đệm thêm cho khối cuối.

Every block is a full 128 MB except the last, which is the remainder. HDFS adds no padding to the last block.

3. Nhân số khối với số bản sao để biết số bản khối phải lưu.
   Multiply the number of blocks by the replication factor to get the number of block copies to store.
4. Nhân dung lượng tệp với số bản sao: đó là dung lượng thực chiếm.
   Multiply the file size by the replication factor: that is the raw storage used.

$$\text{Số khối} = \lceil \text{Dung lượng} \div \text{Kích thước khối} \rceil \tag{2.2}$$

$$\text{Thực chiếm} = \text{Dung lượng} * \text{Số bản sao} \tag{2.3}$$

Ví dụ: tệp nhật ký ứng dụng dài 1.000 MB. Ta có $1.000 \div 128 = 7{,}8$ nên cần 8 khối: 7 khối đầy $7 * 128 = 896$ MB và khối cuối $1.000 - 896 = 104$ MB. Có $8 * 3 = 24$ bản khối. Thực chiếm $3 * 1.000 = 3.000$ MB, tức 3 GB cho 1 GB dữ liệu.

Example: an application log file is 1,000 MB. We get $1{,}000 \div 128 = 7.8$, so 8 blocks are needed: 7 full blocks of $7 * 128 = 896$ MB and a last block of $1{,}000 - 896 = 104$ MB. There are $8 * 3 = 24$ block copies. Raw storage is $3 * 1{,}000 = 3{,}000$ MB, or 3 GB for 1 GB of data.

Đoạn mã sau mô phỏng ba tệp trên cụm 5 nút. Khối thứ i đặt trên nút i, i + 1 và i + 2 (chia vòng). Sau đó ta cho vài nút hỏng và đếm mỗi khối còn bao nhiêu bản. Cách đặt chia vòng này đơn giản hơn HDFS thật, vốn còn tính đến giá máy.

This code simulates three files on a 5-node cluster. Block i is placed on nodes i, i + 1 and i + 2 in rotation. Then we fail a few nodes and count how many copies each block keeps. This round-robin placement is simpler than real HDFS, which also considers racks.

```
KHOI_MB, SO_NUT, BAN_SAO = 128, 5, 3
TEP = {"nhat_ky.log": 1000, "lenh.csv": 300, "sao_ke.json": 90}
khoi = []                          # (tên tệp, số khối, MB)
for ten, mb in TEP.items():
    con_lai, k = mb, 0
    while con_lai > 0:
        khoi.append((ten, k, min(KHOI_MB, con_lai)))
        con_lai -= KHOI_MB
        k += 1
# khối thứ i đặt trên các nút i, i + 1, i + 2 (chia vòng)
giu = [{(i + j) % SO_NUT for j in range(BAN_SAO)}
       for i in range(len(khoi))]
tong = sum(x[2] for x in khoi)
print("số khối:", len(khoi))
print("MB dữ liệu:", tong)
print("MB thực chiếm:", tong * BAN_SAO)
for hong in ({2}, {2, 3}, {0, 1, 2}):
    con = [len(s - hong) for s in giu]
    print(f"hỏng {sorted(hong)}: đủ 3 bản {con.count(3)}, "
          f"còn 2 bản {con.count(2)}, còn 1 bản {con.count(1)}, "
          f"mất {con.count(0)}")
```

Kết quả chạy / *Output*

```
số khối: 12
MB dữ liệu: 1390
MB thực chiếm: 4170
hỏng [2]: đủ 3 bản 4, còn 2 bản 8, còn 1 bản 0, mất 0
hỏng [2, 3]: đủ 3 bản 2, còn 2 bản 5, còn 1 bản 5, mất 0
hỏng [0, 1, 2]: đủ 3 bản 0, còn 2 bản 4, còn 1 bản 5, mất 3
```

Ba tệp cho 12 khối, 1.390 MB dữ liệu và 4.170 MB thực chiếm. Hỏng một nút: 8 khối còn 2 bản, 4 khối còn đủ 3 bản, mất 0 khối. Hỏng hai nút: mất 0 khối, 5 khối chỉ còn 1 bản và sẽ được chép lại ngay. Hỏng ba nút liền nhau: mất 3 khối, vì cả ba bản của chúng nằm đúng ba nút đó.

The three files give 12 blocks, 1,390 MB of data and 4,170 MB of raw storage. With one node down, 8 blocks keep 2 copies, 4 blocks keep all 3, and 0 blocks are lost. With two nodes down, 0 blocks are lost and 5 blocks have a single copy left that will be re-copied at once. With three adjacent nodes down, 3 blocks are lost, because all three of their copies sat on exactly those nodes.

Khái niệm *Concept*	**Bảo vệ khi** *Protects against*	**Không bảo vệ khi** *Does not protect against*
Nhân bản (HDFS) *Replication (HDFS)*	Một hoặc vài máy hỏng *One or a few machines failing*	Xóa nhầm, ghi sai: cả 3 bản bị đổi giống nhau *Accidental delete or bad write: all 3 copies change alike*
Sao lưu *Backup*	Xóa nhầm, ghi sai, mã độc *Accidental delete, bad write, malware*	Hỏng máy giữa hai lần sao lưu *Hardware failing between two backups*

LỖI HAY GẶP / COMMON MISTAKE – Lưu hàng triệu tệp nhỏ trên HDFS. Mỗi tệp và mỗi khối tốn một mẩu bộ nhớ ở nút tên, nên quá nhiều tệp nhỏ làm nút tên quá tải dù tổng dung lượng chỉ vài GB. Gộp tệp nhỏ thành tệp lớn trước khi nạp.
Storing millions of small files on HDFS. Every file and every block costs a piece of memory in the NameNode, so too many small files overload it even when the total is only a few GB. Merge small files into large ones before loading.

2.3. MapReduce: chia việc, xáo trộn, gom kết quả – *2.3. MapReduce: split, shuffle, gather*

Đêm bầu cử, mỗi điểm bầu cử tự đếm phiếu của mình. Kết quả từng điểm được gửi về trung tâm, gom theo ứng viên, rồi cộng lại. Không điểm nào phải đọc phiếu của điểm khác. MapReduce, mô hình do Dean và Ghemawat (2004) mô tả, làm việc theo đúng cách đó.

On election night every polling station counts its own ballots. Each station sends its tallies to the centre, which groups them by candidate and adds them up. No station reads another station's ballots. MapReduce, the model described by Dean and Ghemawat (2004), works exactly this way.

Dữ liệu vào là các dòng, chia thành phân đoạn (split), mỗi phân đoạn đi một máy. Giai đoạn map đọc từng dòng và trả ra cặp khóa – giá trị (một nhãn và một giá trị đi kèm, ví dụ chi nhánh LL và số 1). Giai đoạn xáo trộn và sắp xếp (shuffle and sort) chuyển các cặp cùng khóa về cùng một máy và xếp chúng cạnh nhau. Giai đoạn reduce gộp các giá trị của mỗi khóa thành một kết quả. Máy làm việc gộp này gọi là reducer.

The input is rows, divided into splits, each going to one machine. The map stage reads each row and emits a key-value pair (a label with a value attached, such as branch LL and the number 1). The shuffle and sort stage moves pairs with the same key to the same machine and lines them up. The reduce stage merges the values of each key into one result. The machine that does this merging is called a reducer.

Bài mẫu 2: đếm giao dịch theo chi nhánh – ***Worked example 2: counting transactions by branch***

Ngân hàng Trúc Xanh có chi nhánh đặt theo tên đường: LL (Lê Lợi), HB (Hai Bà Trưng), TH (Trần Hưng Đạo), CG (Cầu Giấy). Ba phân đoạn chứa 12 giao dịch, ghi chi nhánh và số tiền (triệu đồng): phân đoạn 1 gồm (LL, 12), (HB, 5), (LL, 8), (TH, 20); phân đoạn 2 gồm (HB, 7), (LL, 15), (CG, 4), (HB, 9), (LL, 6); phân đoạn 3 gồm (TH, 11), (LL, 10), (HB, 3). Cần đếm số giao dịch mỗi chi nhánh.

Trúc Xanh Bank names its branches after streets: LL (Lê Lợi), HB (Hai Bà Trưng), TH (Trần Hưng Đạo), CG (Cầu Giấy). Three splits hold 12 transactions, listed as branch and amount (million dong): split 1 has (LL, 12), (HB, 5), (LL, 8), (TH, 20); split 2 has (HB, 7), (LL, 15), (CG, 4), (HB, 9), (LL, 6); split 3 has (TH, 11), (LL, 10), (HB, 3). We need to count the transactions of each branch.

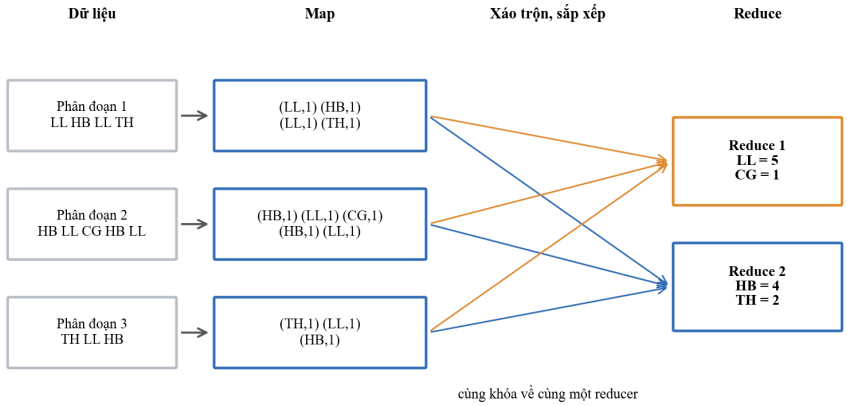
1. Chia dữ liệu thành ba phân đoạn; mỗi phân đoạn đi một máy làm map.
   Divide the data into three splits; each goes to one machine to run map.
2. Hàm map đọc từng dòng và trả cặp (chi nhánh, 1), bỏ qua số tiền.
   The map function reads each row and emits the pair (branch, 1), ignoring the amount.
3. Bộ kết hợp (combiner), nếu có, cộng ngay trong từng máy: (LL,1) hai lần thành (LL,2).
   The combiner, if used, adds inside each machine: (LL,1) twice becomes (LL,2).
4. Xáo trộn: các cặp cùng khóa về cùng reducer. Giả sử Reduce 1 nhận LL và CG, Reduce 2 nhận HB và TH.
   Shuffle: pairs with the same key go to the same reducer. Say Reduce 1 receives LL and CG, and Reduce 2 receives HB and TH.
5. Reduce cộng các giá trị của từng khóa.
   Reduce adds the values of each key.

Phân đoạn *Split*	**Dòng đọc vào** *Rows read*	**Cặp sau map** *Pairs after map*	**Sau bộ kết hợp** *After combiner*
1	LL, HB, LL, TH	(LL,1) (HB,1) (LL,1) (TH,1)	(LL,2) (HB,1) (TH,1)
2	HB, LL, CG, HB, LL	(HB,1) (LL,1) (CG,1) (HB,1) (LL,1)	(HB,2) (LL,2) (CG,1)
3	TH, LL, HB	(TH,1) (LL,1) (HB,1)	(TH,1) (LL,1) (HB,1)

Khóa *Key*	**Giá trị gom được** *Values collected*	**Kết quả reduce** *Reduce result*
LL	2, 2, 1	5
CG	1	1
HB	1, 2, 1	4
TH	1, 1	2

Có 12 giao dịch: LL 5, HB 4, TH 2, CG 1. Không dùng bộ kết hợp, 12 cặp đi qua mạng. Có bộ kết hợp, chỉ 9 cặp đi (3 + 3 + 3). Dữ liệu càng lặp khóa thì khoản tiết kiệm càng lớn.

There are 12 transactions: LL 5, HB 4, TH 2, CG 1. Without a combiner, 12 pairs cross the network. With one, only 9 pairs do (3 + 3 + 3). The more the keys repeat, the larger the saving.



Hình 2.3. Từ trái sang phải: mỗi phân đoạn qua map ra các cặp, xáo trộn đưa cặp cùng khóa về một reducer, reduce cho tổng. Mũi tên cam đi về Reduce 1, mũi tên xanh về Reduce 2.

Figure 2.3. *From left to right: each split goes through map to produce pairs, the shuffle sends same-key pairs to one reducer, and reduce gives the totals. Orange arrows go to Reduce 1, blue arrows to Reduce 2.*

Bộ kết hợp chỉ dùng được khi phép gộp cho cùng kết quả dù gộp theo thứ tự hay nhóm nào. Cộng, đếm, lớn nhất, nhỏ nhất thỏa điều kiện này. Trung bình thì không. Lấy chi nhánh HB với các khoản 5, 7, 9, 3: trung bình đúng là 24 ÷ 4 = 6. Nếu mỗi phân đoạn gửi trung bình riêng (5, 8, 3) rồi lấy trung bình của ba số đó, ta được 5,33, sai. Cách đúng là gửi cặp (đếm, tổng): (1, 5), (2, 16), (1, 3), cộng thành (4, 24) rồi mới chia.

A combiner is valid only when the merge gives the same result however the values are ordered or grouped. Sum, count, maximum and minimum qualify. A mean does not. Take branch HB with amounts 5, 7, 9, 3: the true mean is 24 ÷ 4 = 6. If each split sends its own mean (5, 8, 3) and we average those three, we get 5.33, which is wrong. The right way is to send (count, total) pairs: (1, 5), (2, 16), (1, 3), add them to (4, 24) and only then divide.

$$\text{Trung bình} = \Sigma \text{ tổng} \div \Sigma \text{ đếm} \tag{2.4}$$

Đoạn mã sau chạy đúng các giai đoạn trên 100.000 giao dịch tự sinh, chia 4 phân đoạn. Hàm map trả (1, số tiền), hàm cong cộng hai cặp, nên cùng một hàm dùng được cho bộ kết hợp lẫn reduce.

This code runs the same stages on 100,000 generated transactions in 4 splits. The map function returns (1, amount) and the cong function adds two pairs, so the same function serves both the combiner and reduce.

```
import random
from collections import defaultdict
from functools import reduce
random.seed(7)
CN = ["LL", "HB", "TH", "CG", "TD"]
gd = [(random.choices(CN, [40, 25, 15, 12, 8])[0],
       random.randint(1, 50)) for _ in range(100_000)]
PHAN_DOAN = [gd[i::4] for i in range(4)]
def map_fn(dong):
    cn, tien = dong
    yield cn, (1, tien)            # (đếm, tổng tiền)
def cong(a, b):
    return (a[0] + b[0], a[1] + b[1])
def chay(co_ket_hop):
    qua_mang, gom = 0, defaultdict(list)
    for pd in PHAN_DOAN:           # mỗi phân đoạn một máy
        cuc_bo = defaultdict(list)
        for dong in pd:
            for k, v in map_fn(dong):
                cuc_bo[k].append(v)
        for k, vs in cuc_bo.items():
            if co_ket_hop:
                vs = [reduce(cong, vs)]
            qua_mang += len(vs)
            gom[k].extend(vs)      # xáo trộn: gom theo khóa
    return qua_mang, {k: reduce(cong, v) for k, v in sorted(gom.items())}
a, kq_a = chay(False)
b, kq_b = chay(True)
print("cặp qua mạng không bộ kết hợp:", a)
print("cặp qua mạng có bộ kết hợp:", b)
print("hai cách cùng kết quả:", kq_a == kq_b)
for k, (n, t) in kq_b.items():
    print(f"{k}: {n} giao dịch, tổng {t} triệu, TB {t / n:.2f}")
```

Kết quả chạy / *Output*

```
cặp qua mạng không bộ kết hợp: 100000
cặp qua mạng có bộ kết hợp: 20
hai cách cùng kết quả: True
CG: 11965 giao dịch, tổng 304163 triệu, TB 25.42
HB: 25007 giao dịch, tổng 636898 triệu, TB 25.47
LL: 40126 giao dịch, tổng 1024835 triệu, TB 25.54
TD: 7918 giao dịch, tổng 201432 triệu, TB 25.44
TH: 14984 giao dịch, tổng 381719 triệu, TB 25.48
```

Không bộ kết hợp, 100.000 cặp đi qua mạng; có bộ kết hợp chỉ 20 cặp (4 phân đoạn nhân 5 chi nhánh), giảm 5.000 lần. Hai cách cho cùng kết quả. Chi nhánh LL có 40.126 giao dịch, tổng 1.024.835 triệu đồng, trung bình 25,54 triệu mỗi giao dịch.

Without the combiner, 100,000 pairs cross the network; with it only 20 pairs do (4 splits times 5 branches), a 5,000-fold reduction. Both ways give the same result. Branch LL has 40,126 transactions, a total of 1,024,835 million dong and a mean of 25.54 million per transaction.

Anh Quân giao Bảo Ngọc làm báo cáo đếm giao dịch ví theo chi nhánh mỗi ngày. Chị Thảo hỏi con số lấy từ đâu. Bảo Ngọc trả lời: tổng của bốn phân đoạn, mỗi phân đoạn có số đếm riêng, và bạn đối chiếu lại bằng một câu SQL GROUP BY ở mục 2.11.

Mr Quân asks Bảo Ngọc to produce a daily report counting wallet transactions by branch. Ms Thảo asks where the figure comes from. Bảo Ngọc answers: the total of four splits, each with its own count, cross-checked with a SQL GROUP BY in section 2.11.

> **PHẢI THUỘC / MEMORISE –** MapReduce gồm map (mỗi dòng ra cặp khóa – giá trị), xáo trộn (cùng khóa về cùng nơi) và reduce (gộp giá trị của từng khóa). Bộ kết hợp chỉ dùng cho phép gộp có tính kết hợp và giao hoán: cộng, đếm, lớn nhất. Trung bình phải gửi cặp (đếm, tổng).
>
> *MapReduce consists of map (each row emits a key-value pair), shuffle (same key to the same place) and reduce (merge the values of each key). A combiner is for merges that are associative and commutative: sum, count, maximum. A mean must send (count, total) pairs.*

2.4. Phân vùng dữ liệu và lệch dữ liệu – *2.4. Data partitioning and skew*

Thủ thư xếp sách vào các kệ theo chữ cái đầu của tên tác giả. Kệ chữ N chật cứng, kệ chữ Q gần như trống. Ai cần sách chữ N phải xếp hàng dài. Chia dữ liệu cho nhiều máy cũng gặp đúng chuyện này.

A librarian shelves books by the first letter of the author's name. The N shelf is crammed and the Q shelf is nearly empty. Anyone wanting an N book queues for a long time. Splitting data across machines meets exactly this problem.

Phân vùng (partition) là một phần của bảng được lưu hoặc xử lý riêng. Có hai cách chia phổ biến. Phân vùng theo băm (hash) dùng một hàm biến khóa thành một số rồi lấy phần dư khi chia cho số phân vùng: các khóa phân bố đều nếu khóa có nhiều giá trị khác nhau, nhưng truy vấn theo khoảng phải đọc mọi phân vùng. Phân vùng theo khoảng (range) xếp theo ngày hoặc theo dải số: truy vấn một tuần chỉ cần mở vài phân vùng, xóa dữ liệu cũ cũng dễ, nhưng những ngày đặc biệt như ngày trả lương dồn nhiều giao dịch vào một phân vùng.

A partition is a part of a table stored or processed separately. Two ways of dividing are common. Hash partitioning uses a function that turns a key into a number and takes the remainder when dividing by the number of partitions: keys spread evenly if there are many distinct values, but a range query must read every partition. Range partitioning arranges data by date or by number band: a query over one week opens only a few partitions and deleting old data is easy, but special days such as payday pile many transactions into one partition.

Lệch dữ liệu (data skew) xảy ra khi một phân vùng nhận nhiều hơn hẳn các phân vùng khác. Thủ phạm thường là khóa nóng (hot key), một khóa chiếm tỷ lệ dòng quá lớn, như một sàn thương mại điện tử nhận gần nửa số giao dịch ví. Cả cụm phải chờ nút chậm nhất, nên thêm máy không giúp gì. Ta đo độ lệch bằng tỷ số giữa phân vùng lớn nhất và mức đều.

Data skew occurs when one partition receives far more than the others. The culprit is usually a hot key, one key holding a very large share of rows, such as an e-commerce platform receiving

nearly half the wallet transactions. The whole cluster waits for the slowest node, so adding machines does not help. We measure skew as the ratio of the largest partition to the even level.

$$\text{Độ lệch} = \text{Số dòng lớn nhất} \div (\text{Tổng dòng} \div \text{Số phân vùng}) \qquad (2.5)$$

Độ lệch bằng 1 nghĩa là đều hoàn hảo; bằng 3 nghĩa là phân vùng nặng nhất gấp 3 lần mức đều. Cách chữa phổ biến là thêm muối (salting): gắn hậu tố ngẫu nhiên vào khóa nóng để nó tách thành nhiều khóa con, rải ra nhiều phân vùng. Sau khi tính từng khóa con, ta gộp chúng lại thành một ở bước thứ hai.

A skew of 1 means perfectly even; a skew of 3 means the heaviest partition is 3 times the even level. The usual cure is salting: attach a random suffix to the hot key so that it splits into several sub-keys spread over many partitions. After each sub-key is computed, a second step merges them back into one.

Bài mẫu 3: đo độ lệch bằng tay – ***Worked example 3: measuring skew by hand***

Mười giao dịch có mã điểm bán lần lượt là 4, 4, 4, 4, 4, 4, 4, 5, 9, 2. Chia 3 phân vùng theo quy tắc: phân vùng bằng mã chia dư 3.

Ten transactions have point-of-sale codes 4, 4, 4, 4, 4, 4, 4, 5, 9, 2. Split them into 3 partitions by the rule: partition equals code modulo 3.

1. Mã 4 chia dư 3 được 1, nên 7 dòng vào phân vùng 1. Mã 5 và mã 2 chia dư 2, nên vào phân vùng 2. Mã 9 chia dư 0, vào phân vùng 0.
   Code 4 modulo 3 is 1, so 7 rows go to partition 1. Codes 5 and 2 give remainder 2, so they go to partition 2. Code 9 gives 0 and goes to partition 0.
2. Số dòng ba phân vùng: 1, 7, 2. Mức đều là 10 ÷ 3 = 3,33.
   Rows in the three partitions: 1, 7, 2. The even level is 10 ÷ 3 = 3.33.
3. Độ lệch = 7 ÷ 3,33 = 2,1.
   Skew = 7 ÷ 3.33 = 2.1.
4. Thêm muối: tách bảy dòng mã 4 thành 4a (3 dòng), 4b (2 dòng), 4c (2 dòng), xếp vào phân vùng 0, 1, 2. Số dòng mới: 1 + 3 = 4; 2; 2 + 2 = 4.
   Salting: split the seven rows of code 4 into 4a (3 rows), 4b (2 rows), 4c (2 rows) and place them in partitions 0, 1, 2. New counts: 1 + 3 = 4; 2; 2 + 2 = 4.
5. Độ lệch mới = 4 ÷ 3,33 = 1,2.
   New skew = 4 ÷ 3.33 = 1.2.

Đoạn mã sau làm lại trên 200.000 giao dịch ví, trong đó điểm bán DB000 chiếm 45%, chia 8 phân vùng. Nó thêm một phép đo phân vùng theo ngày trong 30 ngày, với ngày 5 và ngày 25 đông gấp ba.

This code repeats this on 200,000 wallet transactions in which merchant DB000 holds 45%, split into 8 partitions. It adds a measure for date partitions over 30 days, where days 5 and 25 are three times as busy.

```
import hashlib
import random
from collections import Counter
random.seed(11)
SO_PV = 8
DIEM_BAN = [f"DB{i:03d}" for i in range(200)]
gd = [DIEM_BAN[0] if random.random() < 0.45
```

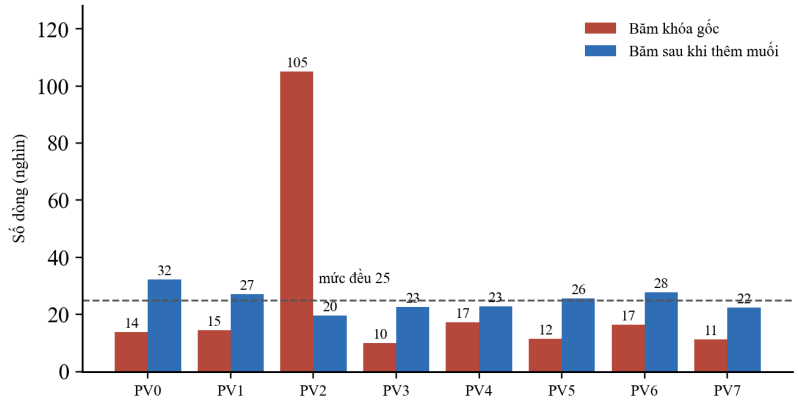
```
      else random.choice(DIEM_BAN[1:]) for _ in range(200_000)]
def bam(khoa):
    return int(hashlib.md5(khoa.encode()).hexdigest(), 16) % SO_PV
def them_muoi(k):
    return f"{k}#{random.randrange(8 * SO_PV)}" if k == "DB000" else k
tho = Counter(bam(k) for k in gd)
muoi = Counter(bam(them_muoi(k)) for k in gd)
TRONG_SO = [3 if d in (5, 25) else 1 for d in range(1, 31)]
ngay = random.choices(range(1, 31), TRONG_SO, k=200_000)
theo_ngay = Counter(ngay)
def do_lech(dem, n):
    return max(dem.values()) / (sum(dem.values()) / n)
print("băm thô:", [tho[i] for i in range(SO_PV)])
print("băm có muối:", [muoi[i] for i in range(SO_PV)])
print("độ lệch băm thô:", round(do_lech(tho, SO_PV), 2))
print("độ lệch băm có muối:", round(do_lech(muoi, SO_PV), 2))
print("độ lệch theo ngày:", round(do_lech(theo_ngay, 30), 2))
```

Kết quả chạy / *Output*

```
băm thô: [13886, 14518, 105116, 10020, 17226, 11501, 16503, 11230]
băm có muối: [32209, 27085, 19636, 22531, 22854, 25642, 27734, 22309]
độ lệch băm thô: 4.2
độ lệch băm có muối: 1.29
độ lệch theo ngày: 2.68
```

Băm khóa gốc cho phân vùng nặng nhất 105.116 dòng trên mức đều 25.000, độ lệch 4,20. Sau khi thêm muối, nặng nhất còn 32.209 dòng, độ lệch 1,29. Chia theo ngày, ngày đông nhất gấp 2,68 lần mức đều của 30 ngày: một dạng lệch khác, không do khóa nóng mà do lịch.

Hashing the original key gives a heaviest partition of 105,116 rows against an even level of 25,000, a skew of 4.20. After salting, the heaviest has 32,209 rows, a skew of 1.29. Partitioning by date, the busiest day is 2.68 times the even level of 30 days: a different kind of skew, caused by the calendar rather than by a hot key.



Hình 2.4. Cột đỏ cao vọt ở một phân vùng là khóa nóng; cột xanh sau khi thêm muối nằm gần đường nét đứt, tức gần mức đều.

Figure 2.4. *The red bar towering in one partition is the hot key; the blue bars after salting sit close to the dashed line, the even level.*

Anh Quân dặn Bảo Ngọc xem độ lệch trước khi chạy bất kỳ công việc nào trên bảng giao dịch ví. Nếu lớn hơn 2, bạn tìm khóa nóng và thêm muối, rồi ghi hai độ lệch trước và sau vào nhật ký công việc để chị Thảo kiểm lại.

Mr Quân tells Bảo Ngọc to check the skew before running any job on the wallet transaction table. If it exceeds 2, find the hot key and add salt, then record the skew before and after in the job log for Ms Thảo to review.

> **LỖI HAY GẶP / COMMON MISTAKE –** Chọn khóa phân vùng chỉ có vài giá trị (loại giao dịch, tỉnh thành) hoặc có khóa nóng. Một phân vùng ôm gần hết việc, các máy còn lại ngồi chờ. Hãy đếm số dòng theo khóa trước khi chọn.
> *Choosing a partition key with only a few values (transaction type, province) or with a hot key. One partition takes nearly all the work while the other machines wait. Count rows by key before choosing.*

2.5. Nối dữ liệu phân tán – *2.5. Distributed joins*

Bảo Ngọc cần gắn vùng (Bắc, Nam, Đông) vào từng giao dịch ví. Giao dịch nằm rải trên nhiều máy, bảng chi nhánh chỉ có vài dòng. Có hai cách ghép. Cách một: đưa cả hai chồng giấy về cùng một bàn theo chữ cái đầu của mã chi nhánh. Cách hai: photo danh bạ chi nhánh nhỏ xíu rồi phát cho mọi bàn, chồng giao dịch đứng yên.

Bảo Ngọc needs to attach a region (North, South, East) to every wallet transaction. The transactions are spread over many machines, while the branch table has only a few rows. There are two ways to match them. One: bring both stacks of paper to the same desk according to the first letter of the branch code. Two: photocopy the tiny branch directory and hand it to every desk while the transaction stack stays where it is.

Cách một gọi là nối xáo trộn (shuffle join): băm khóa nối của cả hai bảng, gửi mỗi dòng đến nút đích, rồi mỗi nút nối phần của mình. Nó dùng được cho hai bảng đều lớn, nhưng gần như mọi dòng đều phải đổi chỗ. Cách hai gọi là nối phát tán (broadcast join): gửi bản sao bảng nhỏ đến mọi nút, mỗi nút nối với phần bảng lớn nó đang giữ. Bảng lớn không di chuyển.

The first way is a shuffle join: hash the join key of both tables, send each row to its destination node, then each node joins its own portion. It works for two large tables, but almost every row has to move. The second is a broadcast join: send a copy of the small table to every node and each node joins it with the part of the large table it already holds. The large table does not move.

Cách nối *Join type*	**Di chuyển** *Movement*	**Điều kiện** *Condition*
Nối xáo trộn *Shuffle join*	Cả hai bảng, gần hết dòng *Both tables, almost every row*	Dùng cho hai bảng lớn *For two large tables*
Nối phát tán *Broadcast join*	Chỉ bảng nhỏ, nhân với số nút *Only the small table, times the node count*	Bảng nhỏ vừa bộ nhớ mỗi nút *The small table fits in each node's memory*

Bài mẫu 4: so sánh khối lượng chuyển – *Worked example 4: comparing the volume moved*

Bảng giao dịch có 1.000.000 dòng trên 50 nút. Bảng chi nhánh có 200 dòng. Tính số dòng phải chuyển qua mạng theo mỗi cách.

The transaction table has 1,000,000 rows on 50 nodes. The branch table has 200 rows. Find how many rows must cross the network under each way.

1. Nối xáo trộn: một dòng giao dịch ở lại nếu nút đích trùng nút đang giữ, xác suất $1 \div 50$. Vậy $49 \div 50$ số dòng phải đi: $1.000.000 * 49 \div 50 = 980.000$ dòng.
   Shuffle join: a transaction row stays only if its destination is the node already holding it, probability $1 \div 50$. So $49 \div 50$ of the rows move: $1,000,000 * 49 \div 50 = 980,000$ rows.
2. Bảng nhỏ cũng bị xáo: $200 * 49 \div 50 = 196$ dòng. Tổng 980.196 dòng.
   The small table is also shuffled: $200 * 49 \div 50 = 196$ rows. The total is 980,196 rows.
3. Nối phát tán: mỗi nút còn lại nhận một bản sao 200 dòng: $(50 - 1) * 200 = 9.800$ dòng.
   Broadcast join: each of the other nodes receives a 200-row copy: $(50 - 1) * 200 = 9,800$ rows.
4. Tỷ lệ: $980.196 \div 9.800 \approx 100$. Nối phát tán chuyển ít hơn khoảng 100 lần.
   Ratio: $980,196 \div 9,800 \approx 100$. The broadcast join moves about 100 times fewer rows.

$$\text{Dòng chuyển} = (n - 1) * \text{Số dòng bảng nhỏ} \tag{2.6}$$

Đoạn mã sau mô phỏng cả hai cách trên 20.000 giao dịch và 4 nút. Nó kiểm tra rằng hai cách cho cùng kết quả: tổng tiền theo vùng.

This code simulates both ways on 20,000 transactions and 4 nodes. It checks that the two ways give the same result: the total amount by region.

```
import hashlib
import random
from collections import defaultdict
random.seed(3)
SO_NUT = 4
VUNG = {"LL": "Bắc", "HB": "Bắc", "TH": "Nam", "CG": "Đông", "TD": "Nam"}
GD = [(i, random.choice(list(VUNG)), random.randint(1, 50))
      for i in range(20_000)]
def nut_dich(khoa):                 # băm khóa nối ra nút đích
    h = hashlib.md5(khoa.encode()).hexdigest()
    return int(h, 16) % SO_NUT
def xao_tron():
    nut = [[] for _ in range(SO_NUT)]
    chuyen = 0
    for d in GD:
        n = nut_dich(d[1])
        nut[n].append(d)
        chuyen += n != d[0] % SO_NUT
    chuyen += sum(nut_dich(c) != 0 for c in VUNG)   # bảng nhỏ ở nút 0
    return chuyen, nut
def phat_tan():
    nut = [[d for d in GD if d[0] % SO_NUT == n] for n in range(SO_NUT)]
    return (SO_NUT - 1) * len(VUNG), nut
def noi(nut):                       # mỗi nút nối với bản bảng nhỏ của nó
    kq = defaultdict(int)
    for dong in nut:
        for _, cn, tien in dong:
            kq[VUNG[cn]] += tien
    return dict(kq)
a, nut_a = xao_tron()
b, nut_b = phat_tan()
print("dòng chuyển nối xáo trộn:", a)
print("dòng chuyển nối phát tán:", b)
print("hai cách cùng kết quả:", noi(nut_a) == noi(nut_b))
print("tổng theo vùng:", noi(nut_b))
```