INF2006 - Big Data Project

In this project, you will develop an end-to-end big data solution that turns raw financial and related data into reproducible analytical results. You will build a traceable data corpus, design a distributed data pipeline, train and evaluate a predictive model, and perform data analytics to answer clearly defined questions. Through these four tasks, you will apply the technologies introduced in the labs and examine how data quality, processing choices and evaluation methods affect the reliability of your findings.

You will act as a financial data engineering team to acquire, organize and process data from multiple sources, and deliver a solution that others can inspect and reproduce. Your team will justify its choice of data, transformations, features and models; use distributed tools to process and analyse the data; and communicate findings supported by queries, model results and visualizations. The emphasis is on sound engineering, careful evaluation and evidence-based interpretation, rather than unsupported claims about investment performance.

Submit one team ZIP by 22 November 2026, 23:59 Singapore time.

Task 1. Build the corpus

1.1 Raw data sources

Use the price snapshot for the core prediction task. Auxiliary inputs remain optional: choose sources that support your question. The reference sizes below describe the available sources; they do not require every team to ingest every source. Freeze and document your selected development and full input profiles before comparing configurations. Any common assessed workload will be confirmed by the teaching team before full runs.

Source	Reference scope and use
Daily price CSV	505 source series; 605,887 accepted bars, 8 February 2013 to 29 December 2017. Daily OHLCV supports cleaning, features and source-price-ratio prediction.
GDELT	731 daily archives, July 2015-June 2017; 144.2 million accepted events. Metadata for distributed parsing and aggregation; 9.61 GB ZIP / 54.71 GB expanded.
SEC EDGAR	311 original 10-K/10-Q filings from 303 issuers across 2013-2017; 2.78 GB. Paragraphs and XBRL facts support document processing.
Wikinews	Two official October 2026 history dumps, about 662 MB compressed. Filter to 2013-2017 article revisions for text and revision analysis.

SEC filing collection

If you select SEC filings, use the supplied fixed inventory of 311 distinct filings from 303 issuers: 156 Form 10-K annual reports and 155 Form 10-Q quarterly reports filed during 2013–2017. You do not need to find another collection or collect filings for all 505 price tickers. Justify any processing or modelling subset.

Download the Excel inventory: SEC filing inventory (.xlsx)

The inventory gives CIK, issuer, form, filing date, accession and SEC URL. Preserve identifiers and leading zeros. It does not verify historical ticker mappings: document relationships used, and check information availability before each prediction.

Use the listed URLs and retain originals, retrieval times, sizes and checksums. Follow SEC access guidance. Report unavailable files; do not silently substitute filings. URL availability has not been rechecked for every filing.

1.2 Requirements

Build a traceable raw-data corpus in HDFS on AWS EMR. Retain original records and distinguish sources, versions and acquisition dates. Justify the storage formats and organization. Task 2 transforms this corpus into analysis-ready data. The Cloud resource constraints section applies to all four tasks; use the labs for setup guidance.

Your corpus should have a documented scope and data dictionary: selected sources and versions, date coverage, native identifiers, schemas, units and time semantics. Preserve source provenance and make supported relationships between tables explicit.

Investigate the raw sources and document coverage, missing or incomplete records, duplicates and corrupted fields. Identify limitations that the data pipeline must address. Preserve original observations and provenance.

Deliver a raw-corpus inventory and reproducibility record containing source references, acquisition times, checksums, configuration and evidence that the same source inputs can be acquired again or identified precisely. Include measured rows/bytes, a quality summary and supporting evidence in the report and its appendices.

Note: In the later tasks, you are **NOT** required to use all datasets, fields or entries.

Task 2. Build the data pipeline

2.1 Required technologies and outcomes

Transform the raw corpus from Task 1 into reproducible analytical tables and features using Hadoop Streaming, HDFS, Hive and PySpark. Use Hadoop Streaming for source processing and audit summaries, Hive for table registration and SQL transformations, and PySpark for distributed cleaning, integration and feature preparation. Each required engine must make a meaningful contribution. Spark SQL alone does not satisfy Hive; financial-table processing must remain distributed rather than collected into pandas.

Explain and justify how the pipeline handles missing values, duplicates, inconsistent or corrupted records and rejected data. Document the cleaning decisions, reconcile source and prepared row counts, and keep raw, cleaned and rejected data distinct.

Check that prices and trading volumes are valid and internally consistent. Preserve genuine gaps and unavailable prediction targets; do not fill them using future observations. Exclude identifiers and information that reveals the target from predictive features.

2.2 Prepare the chronological data splits

As part of data cleaning and preparation, create three chronological datasets: 2013-2015 for training, 2016 for validation, and 2017 for testing. Report the date coverage, row counts and exclusions for each dataset. Do not randomly shuffle observations across these periods.

Prevent data leakage during preparation: construct features from information available at the prediction date, and include a training or validation target only when its full outcome is known within that period. Fit learned transformations, such as imputation and scaling, on training data only and apply them unchanged to validation and test data. Document the treatment of missing values and observations near split boundaries.

2.3 Execution evidence

Provide evidence that the declared jobs and queries executed successfully, with unique IDs and traceable inputs and outputs. Include all referenced mapper/reducer scripts and staged assets. Job configuration and output schemas must be sufficient to reproduce the run.

Ensure Hive tables can be used by Spark and that each pipeline stage has clearly identified inputs and outputs. Document the configuration and storage locations needed to rerun the pipeline.

Task 3. Train and evaluate a model

3.1 Choose a machine learning task

Use the prepared data from Task 2 to define a clear financial prediction question and train a suitable Spark ML model. Specify what you will predict, the prediction horizon and how the result will support your analysis. Choose a task appropriate to the available data; the examples below are options, not a requirement to complete every task.

Possible tasks include: (1) regression to predict the next week's price change; (2) classification to predict whether a price will rise or fall over the next week; and (3) regression or classification to predict future volatility or identify periods of high volatility. As an optional supporting task, classify sentiment in relevant text and investigate whether it improves a price-based model.

Explain how you obtain reliable labels for any text classification task.Define the target and evaluation method clearly. For a weekly price-change task, you may use the change from one trading Friday's opening price to the next, with features available before the first Friday opens. Explain how you handle holidays and missing observations. For any task, use only information that would have been available when making the prediction, and explain how model predictions are converted into the required output weights.

3.2 Training and evaluation

Use the training, validation and test datasets prepared in Task 2. Learn preprocessing and model parameters from the training set, use validation results to select features and settings, then freeze the final approach before evaluating it on the test set. Do not use test results to tune the model. Compare against a simple, clearly defined baseline using suitable metrics on the same observations, and report how unavailable targets are handled.

Compare and justify a manageable set of model or feature choices. If you use auxiliary sources, establish when their information became available and how it relates to the price data. Exclude unsupported inputs and discuss any assumptions or coverage limitations.

3.3 Model reproducibility evidence

Save the complete Spark ML PipelineModel, including learned preprocessing. Demonstrate in a fresh session that loading this saved model reproduces its predictions without retraining. Report any differences using a stated numerical tolerance, and clearly flag missing or incompatible model files.

Task 4. Perform data analytics

4.1 Analytical questions and findings

Use Hive and/or PySpark to analyse the prepared data and model outputs. Answer clear analytical questions using appropriate queries, aggregations and comparisons. Explain why the chosen tools suit each analysis.

Present visualizations, interpret the findings, and support conclusions with evidence. Examine patterns, differences and model behaviour relevant to the project question; distinguish observed relationships from causal claims. Discuss data limitations and uncertainty. Descriptive analysis may also inform earlier corpus, pipeline and modelling decisions.

Keep processing of the full data distributed. Plotting libraries may visualize suitably sized aggregated results without replacing Hive or PySpark processing.

4.2 Example business questions

The following examples are options; you are not required to complete all of them. Choose questions that support your project objective and available data. Use Hive and/or PySpark for the analysis.

(1) When is trading activity unusually high or low? Compare trading volume with each series' own historical pattern, then examine how unusual activity relates to price movements. Use grouped summaries, trend charts or scatter plots. Do not assume raw volume is directly comparable across different securities.

(2) Do news or event signals help explain market behaviour? Where timing and links to price series are supported, compare subsequent price changes or volatility across news-volume or sentiment groups. Present group sizes and distributions, and discuss coverage and alternative explanations; association alone does not establish causation.

(3) Where is the predictive model most and least reliable? Compare held-out model errors or classification metrics with the baseline across time periods and volatility groups. Show subgroup sample sizes and an error chart or confusion matrix. Use the findings to discuss deployment limitations without retuning on test results.

4.3 Required evidence

Provide reproducible SQL queries and/or PySpark notebook code, identified inputs, query outputs or result tables, and clearly labelled visualizations. Link each conclusion to the analysis that supports it. Compare model results with the Task 3 baseline and explain their practical meaning and limitations.

Report source-price-ratio/model diagnostics without treating them as economic returns or proof of historical investability. Include the scope, row counts and relevant execution evidence needed to reproduce the analysis.

Cloud resource constraints

Use one primary and two core m5.xlarge EMR nodes, with no task nodes or autoscaling. Local development is allowed, but it does not qualify as evidence of the assessed EMR run. The following common reference configuration is provisional until checked in AWS Academy. The teaching team will confirm the release, permitted adjustments and assessed workload before full runs; develop on a small sample until then.

Setting	Provisional reference
EMR release	emr-7.10.0
Cluster	One primary and two core m5.xlarge nodes; no task nodes or autoscaling
Worker data storage	One 200 GiB gp3 data volume per core node; 3,000 IOPS and 125 MiB/s per volume
HDFS replication	Explicitly set dfs.replication to 2
Spark starting allocation	Two executors, each with two cores and 6 GiB heap; dynamic allocation off
Memory allowance	Plan for executor overhead and driver/application-master capacity; the proposal allows 2 GiB overhead per executor and 2 GiB driver heap, subject to the course pilot

Estimate storage and memory before a full run. The two worker data volumes provide 400 GiB physical storage, with a nominal 200 GiB logical HDFS ceiling at replication 2 before overhead and working space. Budget raw and prepared files, simultaneous copies, shuffle, temporary files and logs; measure peak use and free space on each worker. Do not add primary or root storage to worker HDFS capacity, or assume that compressed and expanded source sizes are interchangeable.

Compare the reference configuration with one justified alternative using identical input manifests and output checks. Propose changes to executor allocation, partitions, joins, intermediate storage or data-volume size, and justify their expected effect. The teaching team will confirm permitted adjustments before assessed runs. Keep hardware, release, replication and storage performance fixed when testing software changes; evaluate storage-size changes separately.

Runtime comparison

Start timing after the original raw inputs are staged in HDFS. Exclude downloading, S3-to-HDFS copying, dependency setup and cluster startup, and report those times separately. Include parsing or decompression needed from the staged inputs, Hadoop Streaming, Hive transformations, distributed feature preparation, training and required output writes and checks. Do not reuse prepared features or fitted models to bypass the timed work.

Compare development and full profiles using genuine input growth. Record all three repetitions for each compared profile/configuration, their median and range, stage timings, actual bytes and rows, resource settings, measured memory and disk use, and failures. Use fresh application state and output locations; keep cache treatment consistent. Explain correctness and performance trade-offs using execution evidence. No speedup is a valid finding; faster runtime alone earns no higher marks. These comparisons use the existing performance and reproducibility criterion, with no change to rubric weights.

The 60-minute marking limit is provisional. It will be enforced only after the teaching team has checked that the confirmed workload can reasonably meet it on the issued configuration and has communicated the final timing policy. The proposed configuration has not yet been launched or timed in AWS Academy.

Report requirements

Submit a report of at most 3,000 words in the main text, excluding the cover, references, figure/table captions and appendices. Use the required numbered headings below, in this order. You may add subheadings. Keep essential analysis and key results in the main report; replace the guidance with your own analysis.

Cover

State the project title, assigned team code, every member's name and student ID, and submission date.

1. Project overview and objectives

Define the business problem, intended users, analytical questions and project scope. Include a labelled diagram showing the end-to-end data flow and technologies.

2. Building the corpus (Task 1)

Describe selected sources, provenance, coverage, schemas and storage choices. Summarize the corpus inventory and raw-data quality findings.

3. Building the data pipeline (Task 2)

Explain cleaning, integration and feature preparation; justify the contributions of Hadoop Streaming, HDFS, Hive and PySpark. Describe chronological splits and leakage controls, with execution evidence.

4. Model training and evaluation (Task 3)

Define the prediction task, features, baseline and metrics. Compare experiments, justify the final model, present held-out results and demonstrate saved-model reproduction.

5. Data analytics and business findings (Task 4)

State each business question and the corresponding analysis. Present labelled visualizations, interpret results and discuss implications and limitations.

6. Performance and scalability

Report stage and end-to-end timings, data sizes, storage estimates and measured memory/disk use. Compare development and full workloads, and the reference configuration against your justified alternative on identical inputs. Report all repetitions and medians, verify outputs, explain bottlenecks and discuss trade-offs.

7. Conclusions and limitations

Answer the project questions, summarize achievements and limitations, and propose realistic improvements.

References

Cite datasets, methods, software and reused materials, including relevant versions.

Required appendices

Appendix A: EMR setup and rerun guide (following the requirements below).

Appendix B: supporting evidence with file, notebook-cell or job locators, and

Appendix C: Use of AI assistance, such as tools, purpose and affected work.

Appendix D: team contributions, using the table shown in the Submission requirements section.

Required EMR setup and rerun guide

Include the following as Appendix A of the report. Provide clear, numbered instructions specific to your submitted project so that the teaching team can set it up and rerun it on a fresh AWS EMR environment. A reference to the labs alone is not sufficient. You may provide commands or configuration files in the ZIP and link to them by relative path.

1. Prerequisites and cluster configuration

State the EMR release, required applications and software versions, permitted cluster configuration, and required AWS permissions and storage access. Identify configurable resource names and paths; never include credentials.

2. Prepare the environment

Explain how to connect to the cluster, install or load dependencies, configure the notebook environment, and make Hadoop, Hive and Spark available. Identify any bootstrap scripts or configuration files and their execution order.

3. Acquire and stage the data

Give source versions and access instructions, download or retrieval steps, expected sizes or counts, and the process for loading data into HDFS. Explain Hive table creation and access from Spark.

4. Upload and execute the project

Identify the exported .ipynb filename, helper scripts and configuration files. Explain how to upload/open the notebook in EMR, set paths and parameters, and run each pipeline, training and analytics stage in order. State expected outputs and approximate runtimes.

5. Reload the saved model and verify results

Give the checkpoint location and steps to load it in a fresh session without retraining. Describe the checks and expected results that confirm the pipeline, predictions and analytics have reproduced successfully. Include dated evidence of your own fresh-session rerun.

6. Troubleshooting and cleanup

Explain common setup or execution failures and how to diagnose them. Describe how to preserve required outputs and evidence, then terminate the cluster and remove temporary resources to avoid unnecessary charges.

Assessment rubric (100 marks)

Criterion	Mark	What earns strong marks	Evidence
Problem framing and design	10	Clear business questions, justified scope and a coherent end-to-end design.	Report section 1; labelled data-flow diagram.
Corpus quality and provenance (Task 1)	15	Relevant, traceable sources; documented schemas and coverage; sound raw-data quality assessment.	Report section 2; corpus inventory, data dictionary and source records.
Distributed data pipeline (Task 2)	20	Working, reproducible processing with meaningful use of all required engines; justified cleaning, integration and leakage controls.	Report section 3; notebook, scripts, Hive queries, job records and prepared-data checks.
Model training and evaluation (Task 3)	20	Well-defined target; appropriate baseline and metrics; fair chronological evaluation; justified experiments and reproducible saved model.	Report section 4; experiment results, test metrics, complete checkpoint and reload evidence.
Data analytics and interpretation (Task 4)	15	Relevant questions answered with correct Hive/PySpark analyses; clear visualizations, supported conclusions and thoughtful limitations.	Report section 5; queries, result tables and labelled visualizations.
EMR execution, performance and reproducibility	15	Clear setup and rerun instructions; verified EMR execution; credible timing/scaling analysis and complete reproducibility records.	Report section 6 and Appendix A; dated EMR evidence, runtime records and setup instructions.
Communication, contribution and attribution	5	Clear, concise report using the required headings; traceable evidence, specific contributions and transparent attribution.	Report, Appendices B and C, and submission file list.

Marker principle: marks are awarded for verifiable evidence and sound reasoning. Unsupported claims or screenshots alone do not establish correct execution. Missing or inaccessible evidence earns no credit for the associated claim.

Use the required report headings and identify evidence by exact filename and relative path. The teaching team must be able to follow Appendix A and rerun the submitted notebook. Retain configuration and dated execution evidence; you do not need to keep an EMR cluster running after completing the work.

Submission requirements

Submit one team ZIP. Before submission, confirm that:

- The ZIP contains report.pdf, the actual .ipynb notebook exported from EMR, the ML model(s), and all required helper scripts and assets.
- The report uses the required headings, includes all appendices, and stays within the 3,000-word main-text limit.
- Appendix A gives clear EMR setup, data access, dependency, configuration and execution steps, including the notebook filename and run order.
- The exported notebook has been rerun in a fresh EMR session, and the saved model reproduces predictions without retraining.
- Each verification record states the check, steps or command, expected result, actual result and evidence location; failed checks are reported honestly.
- Appendix A records the data versions, training cutoff, software versions, random seeds and saved-model locations needed to reproduce the project.
- Every referenced submission file exists and opens using a path relative to the ZIP root. State the main-report word count on the report cover and use clear page numbers and appendix headings.
- The submission contains no credentials, access tokens, raw datasets or machine-specific paths.
- Data sources, reused code and AI assistance are declared, with the team's verification recorded in Appendix B.
- Appendix C lists every member's work delivered and estimated contribution percentage; the percentages total 100%.
- A teammate can follow Appendix A to set up the project, run the notebook and locate all required evidence.

Required Appendix C: team contributions

Include the following table in the report, with one row per member. Describe concrete work delivered and agree on estimated shares of the total team contribution. The percentages must total 100%. Add or remove rows to match your team size.

Member name	Work delivered	Est. contribution %