Exercises

MGMT 675: Generative AI for Finance

00 – Course Introduction

Exercise 1: AI Use Cases in Finance. Pick a specific finance role (e.g., equity research analyst, credit underwriter, portfolio manager, compliance officer). Using Claude.ai, have a conversation in which you identify five concrete tasks that AI could automate or augment in that role. For each task, classify it as "maker" work or "checker" work (per the JP Morgan framing in the slides). Submit a one-page summary with your classifications and a brief argument for which task would deliver the most value if automated first.

Exercise 2: AI Capabilities Audit. Choose a publicly traded company and ask Claude to (a) summarize its most recent earnings release, (b) pull its current P/E ratio from the web, and (c) generate a simple bar chart comparing its revenue over the past four quarters. For each task, note which AI capability from the slides it relies on (e.g., document analysis, web search, code execution). Submit screenshots of the conversation along with your capability mapping.

Exercise 3: Cross-Model Evaluation. Ask Claude and at least one other AI (ChatGPT, Gemini, or Copilot) the same finance question: *"Explain the Modigliani-Miller theorem and its real-world limitations."* Compare the two responses on accuracy, depth, and clarity. Then start a new Claude conversation and ask it to critique the other model's answer. Submit all three outputs with a short reflection (half page) on what you learned about cross-evaluation.

01 – Claude.ai / Desktop

Exercise 4: Interactive Bond Pricing App. Using Claude.ai Artifacts, create an interactive app that lets a user input a bond's coupon rate, yield to maturity, face value, and years to maturity (assume semi-annual coupons). The app should compute the bond price and display an interactive chart showing how the price changes as the yield varies from 0% to 15%. Publish the artifact and submit the shareable link.

Exercise 5: Iterative Prompting for a Financial Model. Ask Claude.ai to create an Excel file that models a leveraged buyout (LBO) for a hypothetical company. Start with a rough prompt, then refine iteratively over at least four turns (e.g., add debt schedules, adjust assumptions, improve formatting). Keep a log of each prompt you sent and what changed in the output. Submit the final Excel file and your prompt log with a brief reflection on the plan-execute-evaluate cycle.

Exercise 6: System Prompt Design. Write a system prompt (300–500 words) for a custom chatbot that serves as a "client portfolio advisor." The prompt should define the chatbot's persona, specify its areas of expertise, set guardrails (e.g., never give specific buy/sell recommendations), and describe the response format. Test your system prompt by pasting it into a new Claude conversation as the first message, then ask three client-style questions. Submit the system prompt and the three Q&A exchanges.

02 – Mean-Variance Optimization

Exercise 7: Tangency Portfolio with Real Data. Download monthly return data for five ETFs of your choice (e.g., SPY, AGG, GLD, VNQ, EFA) and the one-month T-bill rate from a source such as FRED or Yahoo Finance. Upload the data to Claude and ask it to compute the tangency portfolio weights, the expected return and standard deviation of the tangency portfolio, and the Sharpe ratio. Have Claude plot the efficient frontier and the capital allocation line. Submit the Excel data file, the Claude conversation, and the final plot.

Exercise 8: Constrained Optimization Comparison. Using the same data from Exercise 7 (or a new set of at least four assets), ask Claude to solve for the tangency portfolio under three scenarios: (a) unconstrained, (b) no short sales, and (c) maximum 40% in any single asset. Compare the portfolio weights, Sharpe ratios, and efficient frontiers across all three cases. Submit a one-page memo discussing how constraints affect diversification and performance.

Exercise 9: Solver Comparison. Ask Claude to find the tangency portfolio for a set of at least three assets using two different Python solvers (e.g., scipy.minimize and cvxpy). Compare the results: Are the portfolio weights identical? How do computation times compare? Under what circumstances might one solver be preferred? Submit the code, the outputs, and a half-page discussion.

03 – Google Gemini in Colab

Exercise 10: Stock Return Analysis in Colab. In Google Colab, use Gemini to write code that downloads daily price data for three stocks of your choice from Yahoo Finance, computes daily returns, and produces (a) a boxplot of the daily returns, (b) a summary statistics table (mean, std, min, max, skewness, kurtosis), and (c) a correlation matrix heatmap. Save the notebook and the output files to Google Drive. Submit the notebook (.ipynb) and a screenshot of the outputs.

Exercise 11: Beta Estimation. Download betas.xlsx to your Google Drive. In Colab, ask Gemini to mount your Drive, load the data, estimate betas for each stock using OLS regression, and create a bar chart of the estimated betas with 95% confidence intervals. Ask Gemini to interpret the results: which stocks are defensive and which are aggressive? Submit the notebook and a short written summary.

Exercise 12: Colab vs. Claude Comparison. Perform the same analysis task in both Google Colab (with Gemini) and Claude.ai: compute the annualized mean return and volatility for a set of at least three stocks using one year of daily data. Compare the two experiences: Which was easier to set up? Which produced more reliable code on the first attempt? Which gave better explanations? Submit both outputs and a one-page comparison.

04 – Model Context Protocol

Exercise 13: Financial Data with MCP. Install the Alpha Vantage or Financial Modeling Prep MCP server in Claude Desktop. Use it to retrieve financial data for a company of your choice (e.g., income statement, balance sheet, or stock prices). Ask Claude to analyze the data and produce a summary table in Excel. Submit screenshots showing (a) the MCP server in your config, (b) the hammer icon with available tools, and (c) the final Excel output.

Exercise 14: Browser Automation for Data Collection. Install the Browser-Use MCP server in Claude Desktop. Ask Claude to navigate to a financial website (e.g., Yahoo Finance or the SEC EDGAR site), extract a specific piece of data (e.g., the current market cap and P/E ratio for a company), and present it in a formatted response. Submit screenshots of the browser automation in action and the final result.

Exercise 15: Multi-Server Workflow. Configure at least two MCP servers in Claude Desktop (e.g., DesktopCommander + a financial data server). Design a workflow where Claude uses both servers in a single conversation—for example, pulling financial data via the data server and then running a Python script via DesktopCommander to analyze it. Submit your config file (with API keys redacted), the conversation transcript, and the output.

05 – Claude Cowork

Exercise 16: Portfolio Performance Report. Create a CSV file containing monthly returns for at least three funds over the past two years. Place it in a folder on your computer. In Claude Desktop's Cowork tab, point Claude at the folder and ask it to compute the Sharpe ratio for each fund, identify the top performer, and create an Excel spreadsheet with a summary table and a bar chart of the Sharpe ratios. Submit the CSV input, the Excel output, and a screenshot of the Cowork session.

Exercise 17: Multi-File Document Processing. Download exercise2_invoices.zip and extract it into a folder. In Cowork, ask Claude to read the invoice register (PDF), the payment log (xlsx), and the vendor disputes memo (docx). Have Claude match invoices to payments, apply dispute resolutions, and produce a reconciliation summary with the total outstanding amount. Submit the input files and the reconciliation output.

Exercise 18: Cowork vs. Code Mode Comparison. Perform the same data analysis task (e.g., computing descriptive statistics and a histogram for a dataset) in both the Cowork tab and the Code (Local) tab. Compare: Which was faster? Which used more tokens (check with /cost)? Which gave you more control over the output? Write a half-page reflection on when you would use each mode.

06 – Claude Desktop in Code Mode

Exercise 19: Fama-French Regression. Download or create a CSV file with monthly returns for five industry portfolios and the Fama-French three factors (Mkt-RF, SMB, HML) plus RF. In Claude Desktop's Code tab (Local), ask Claude to run a Fama-French three-factor regression for each portfolio, save the results (alpha, betas, R-squared) to an Excel file, and create a plot of the estimated market betas with confidence intervals. Submit the data file, the Excel output, and the plot.

Exercise 20: Invoice Reconciliation in Code Mode. Download exercise2_invoices.zip into your project folder. In Code (Local), ask Claude to extract the invoice table from the PDF, match invoices to payments from the Excel file (handling inconsistent reference formats), identify fully paid, partially paid, and unpaid invoices, and apply dispute resolutions from the Word memo. Ask for a final reconciliation summary. Submit the output and a brief description of the steps Claude took.

Exercise 21: Subagent Design. In Claude Code, use the `/agents` command to create a custom subagent called "financial-analyst" that is specialized for analyzing CSV data files. Define its system prompt to include standards such as always annualizing returns, labeling chart axes, and saving results to Excel. Test the subagent on a sample dataset. Submit the agent's markdown file (`.claude/agents/analyst.md`), the test conversation, and the output.

07 – Claude Add-In for Excel

Exercise 22: Two-Stage DCF in Excel. Open a blank Excel workbook and use the Claude Excel Add-in to build a two-stage DCF model. The model should include (a) explicit forecast period assumptions (revenue growth, margins, capex, etc.), (b) a terminal value calculation, (c) a WACC computation, and (d) a final enterprise value and implied share price. Ensure all calculations use Excel formulas, not hardcoded values. Submit the workbook.

Exercise 23: Beta Estimation in Excel. Download betas.xlsx and open it in Excel with the Claude Add-in. Ask Claude to compute excess returns for WMT and run a regression to estimate its beta. Then ask Claude to create a scatter plot of excess returns with the regression line displayed. Submit the workbook with the regression results and chart.

Exercise 24: Debugging a Broken Spreadsheet. Obtain or create an Excel workbook that contains several intentional errors: a #REF! error, a circular reference, a #DIV/0! error, and at least one hardcoded value that should be a formula. Open it with the Claude Add-in and ask Claude to identify and fix all errors. Submit the original (broken) workbook and the corrected version, along with Claude's explanation of each fix.

08 – Claude Skills

Exercise 25: Investment Memo Skill. Create a Claude skill called `investment-memo` in your project's `.claude/skills/` folder. The SKILL.md file should define the output format (executive summary, business overview, financial analysis, risk factors, recommendation), specify workflows, and include code examples. Test the skill by asking Claude to write an investment memo for a public company. Submit the SKILL.md file and the generated memo.

Exercise 26: Portfolio Report Skill. Create a skill called `portfolio-report` that instructs Claude to generate a monthly portfolio performance report. The skill should specify the report structure (holdings summary, performance attribution, risk metrics, charts), require annualized returns and Sharpe ratios, and include a Python script template for generating charts. Test it with sample data. Submit the skill folder and the generated report.

Exercise 27: Skill vs. Direct Prompting Comparison. Pick a recurring financial analysis task (e.g., analyzing a quarterly earnings release). First, perform the task by prompting Claude directly with detailed instructions. Then, create a skill that encodes those instructions and perform the same task using the skill. Compare the two outputs on consistency, quality, and effort. Submit both outputs and a half-page reflection on when skills add value over direct prompting.

09 – Claude Code in VS Code

Exercise 28: Beta Estimation with Word Report. In VS Code with Claude Code, open a folder containing betas.xlsx. Ask Claude Code to compute WMT's excess returns, run a regression to estimate its beta, and produce a Word document containing a scatter plot of the data with the regression line and a discussion of why the beta is what it is. Submit the Word document and the Python script Claude wrote.

Exercise 29: Table Aggregation. Download aggregation.zip into your VS Code project folder and extract the workbooks. Each workbook contains a table with similar but not identical columns and column names. Ask Claude Code to combine all tables into a single unified table, reconciling the varying column names and including all columns. Submit the combined table (as Excel or CSV) and a description of how Claude handled the name reconciliation.

Exercise 30: Jupyter Notebook Analysis. Ask Claude Code to create a Jupyter notebook that (a) loads a financial dataset of your choice, (b) computes summary statistics, (c) creates at least two visualizations (e.g., time series plot and histogram), and (d) includes markdown cells explaining each step. Run the notebook and verify all cells execute without errors. Submit the completed notebook (.ipynb).

10 – Discounted Cash Flow Analysis

Exercise 31: Full DCF Valuation. Choose a publicly traded company. Ask Claude to build a complete DCF valuation using publicly available financial data. The analysis should include (a) a five-year revenue and free cash flow projection, (b) a terminal value estimate using both the perpetuity growth method and the exit multiple method, (c) a WACC calculation, and (d) a sensitivity table showing implied share prices under different WACC and growth rate assumptions. Submit the analysis as an Excel workbook with all formulas intact.

Exercise 32: DCF Cross-Check. Perform a DCF valuation for a company using Claude, then start a new conversation and ask Claude to critique the valuation. Have the second conversation identify at least three questionable assumptions and suggest alternatives. Revise the model based on the critique. Submit the original valuation, the critique, and the revised valuation with a summary of changes.

Exercise 33: Comparable Companies Analysis. Ask Claude to build a comparable companies (comps) analysis for a target company. The analysis should identify five peer companies, pull relevant multiples (EV/EBITDA, P/E, EV/Revenue), compute the median and mean multiples, and apply them to the target's financials to derive an implied valuation range. Present the results in a formatted Excel workbook. Submit the workbook and a half-page memo discussing which multiples are most relevant for the industry.

11 – Retrieval Augmented Generation

Exercise 34: NotebookLM Financial Analysis. Upload at least three financial documents (e.g., a 10-K filing, an earnings call transcript, and an analyst report) for the same company into Google NotebookLM. Ask at least five questions about the company's financial performance, strategy, and risk factors. Note how NotebookLM cites specific sources in its answers. Submit the five Q&A pairs with citations and a brief assessment of answer quality.

Exercise 35: RAG Pipeline in Colab. Work through the Agentic RAG for Dummies notebook using a corporate annual report of your choice (not Walmart). Ask at least five finance-specific questions (e.g., revenue breakdown, risk factors, capital allocation plans). For each question, note whether the RAG system's answer is grounded in the document or appears to be hallucinated. Submit the notebook and your evaluation of the five answers.

Exercise 36: RAG vs. Direct Prompting. Choose a financial document (e.g., a 10-K filing). Ask Claude the same five questions about the document in two ways: (a) upload the document directly to Claude.ai and ask, (b) use NotebookLM or the RAG Colab notebook. Compare the answers on accuracy, level of detail, and citation quality. Submit all ten answers and a one-page comparison discussing when RAG adds value over direct context injection.

12 – Fine-Tuning and Small Language Models

Exercise 37: Train nanoGPT on Shakespeare. Follow the instructions in the slides to train a character-level GPT on Shakespeare's complete works using Karpathy's nanoGPT. Run the training on CPU with the suggested hyperparameters. After training, generate a sample of pseudo-Shakespearean text. Submit (a) a screenshot of the training loss over iterations, (b) a generated text sample (at least 500 characters), and (c) a half-page reflection on what the model learned and where it fails.

Exercise 38: GPT from Scratch Notebook. Work through the GPT from Scratch notebook. For each major section (tokenization, positional encoding, self-attention, transformer blocks, training loop, inference), write a one-paragraph explanation in your own words of what is happening and why. Submit the completed notebook with your explanations added as markdown cells.

Exercise 39: RAG vs. Fine-Tuning Analysis. Write a one-page memo addressed to the CTO of a hypothetical asset management firm that is considering how to give its internal chatbot access to proprietary research reports. Compare three approaches: (a) RAG, (b) fine-tuning a pre-trained model, and (c) training a small language model. For each approach, discuss setup effort, data requirements, cost, privacy implications, and suitability for the firm's needs. Recommend one approach and justify your choice.

13 – Building an AI Agent

Exercise 40: Simple Chatbot. Using Python and the OpenRouter API (or the Anthropic API), build a simple chatbot that (a) accepts user input in a loop, (b) maintains conversation history, (c) includes a finance-focused system prompt, and (d) displays responses. The chatbot should run in the terminal. Test it with at least five finance questions. Submit the Python script and a transcript of the test conversation.

Exercise 41: Database Agent. Using Claude Code or a Python script, build an agent that has access to a SQL tool (using SQLite). Create a small SQLite database with at least two tables of financial data (e.g., stock prices and company fundamentals). Write a system prompt that describes the database schema. Implement the agent loop so the LLM can write SQL, execute it, see the results, and respond to the user. Submit the code, the database file, and a sample conversation.

Exercise 42: Agent with Streamlit UI. Extend either Exercise 40 or 41 by adding a Streamlit web interface. The app should display a chat window where the user can type questions and see the agent's responses. If using the database agent, display query results in a formatted table. Deploy the app locally and submit the code, a screenshot of the running app, and a brief description of how the agent loop integrates with Streamlit.

14 – Trading on News with AI

Exercise 43: Headline Sentiment Classification. Collect 20 recent financial news headlines (from sources like Reuters, Bloomberg, or Yahoo Finance). Using Claude or the OpenRouter API, classify each headline as positive, negative, or neutral for the stock mentioned. Also have Claude rate the magnitude (1–5) and relevance (low/medium/high). Compare Claude's classifications with the actual stock price movement on the day of the headline. Submit the 20 headlines, Claude's classifications, the actual price movements, and a summary of Claude's accuracy.

Exercise 44: FinBERT vs. LLM Comparison. Using Google Colab, run FinBERT (from HuggingFace's `ProsusAI/finbert` model) on 10 financial headlines. Then classify the same headlines using an LLM (Claude or GPT via the API). Compare the two models' sentiment predictions on accuracy, nuance, and speed. Identify at least two headlines where the models disagree and explain which model's classification is more reasonable. Submit the notebook, the comparison table, and your analysis.

Exercise 45: Earnings Call Tone Analysis. Find a recent earnings call transcript for a publicly traded company (available from Seeking Alpha, company investor relations pages, or SEC filings). Ask Claude to analyze the management's tone on a scale from -5 (very bearish) to +5 (very bullish), identify forward-looking statements that differ from consensus expectations, and compare the current quarter's tone to a prior quarter. Submit the transcript (or a link), Claude's analysis, and a brief assessment of whether the tone analysis aligns with the stock's subsequent performance.