2024 Fall Final report: Portfolio Team

Part 1

Basic portfolio construction:

Initialization:

- Accepts `tickers` (stock symbols) and `total_investment` as inputs.
- Retrieves market capitalization, current prices, and historical data for the stocks.
- Filters stocks based on constraints like minimum market capitalization.

Weight Calculation:

- Implements 11 weighting strategies:
  - **Market Cap Weighted**: Allocates based on the relative size of each company.
  - **Equal Weighted**: Assigns equal weights to all stocks.
  - **Score Tilt Weighted**: Uses ranked historical returns to allocate weights.
  - **Score Weighted**: Allocates based on volatility scores, prioritizing less volatile stocks.
  - **Inverse Volatility Weighted**: Gives higher weight to stocks with lower volatility.
  - **Minimum Correlation Weighted**: Allocates weights to minimize overall portfolio correlation.
  - **Dividend Yield Weighted**: Allocates based on the relative dividend yield of stocks.
  - **Momentum Weighted**: Uses historical price momentum to determine allocations.
  - **Earnings Yield Weighted**: Allocates based on earnings-to-market cap ratio.
  - **Volatility Trend Weighted**: Focuses on rolling volatility trends for allocation.
  - **Value Weighted**: Allocates using a P/E ratio-derived value approach.

Portfolio Construction:

- Initializes positions based on combined weights and calculates shares to buy.

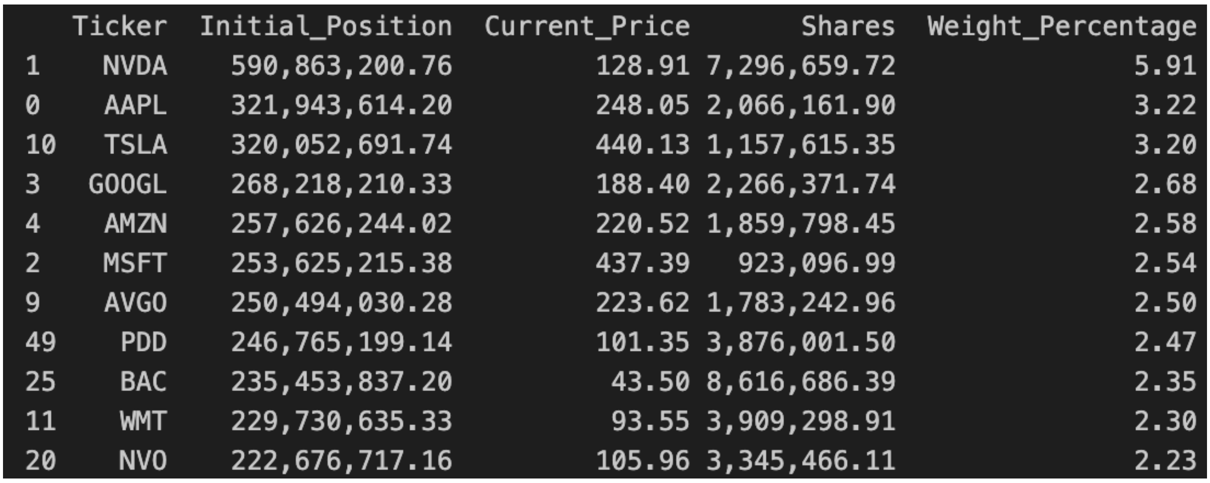
Rebalancing:

- Simulates daily portfolio rebalancing, tracking trades, total value, and returns.

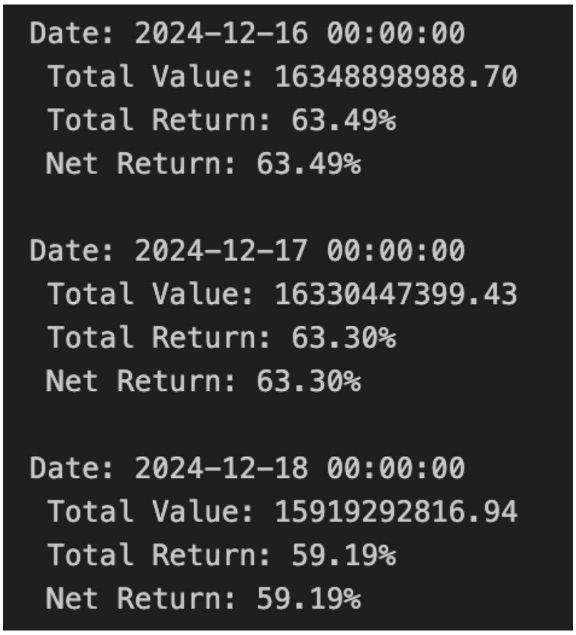
Data Management:

- Retrieves detailed financial data for each stock, including P/E ratio, beta, dividends, and historical trends.

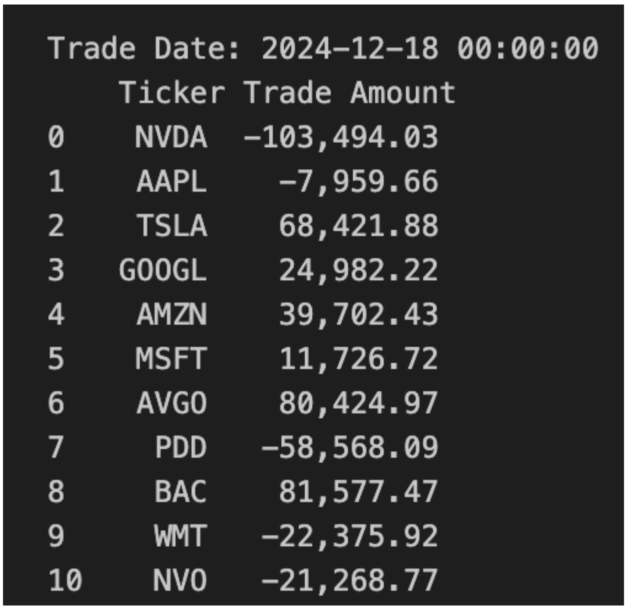
Results



```
   Ticker  Initial_Position  Current_Price       Shares  Weight_Percentage
1    NVDA    590,863,200.76         128.91 7,296,659.72               5.91
0    AAPL    321,943,614.20         248.05 2,066,161.90               3.22
10   TSLA    320,052,691.74         440.13 1,157,615.35               3.20
3   GOOGL    268,218,210.33         188.40 2,266,371.74               2.68
4    AMZN    257,626,244.02         220.52 1,859,798.45               2.58
2    MSFT    253,625,215.38         437.39   923,096.99               2.54
9    AVGO    250,494,030.28         223.62 1,783,242.96               2.50
49    PDD    246,765,199.14         101.35 3,876,001.50               2.47
25    BAC    235,453,837.20          43.50 8,616,686.39               2.35
11    WMT    229,730,635.33          93.55 3,909,298.91               2.30
20    NVO    222,676,717.16         105.96 3,345,466.11               2.23
```



```
Date: 2024-12-16 00:00:00
 Total Value: 16348898988.70
 Total Return: 63.49%
 Net Return: 63.49%

Date: 2024-12-17 00:00:00
 Total Value: 16330447399.43
 Total Return: 63.30%
 Net Return: 63.30%

Date: 2024-12-18 00:00:00
 Total Value: 15919292816.94
 Total Return: 59.19%
 Net Return: 59.19%
```



```
Trade Date: 2024-12-18 00:00:00
    Ticker Trade Amount
0     NVDA  -103,494.03
1     AAPL    -7,959.66
2     TSLA    68,421.88
3    GOOGL    24,982.22
4     AMZN    39,702.43
5     MSFT    11,726.72
6     AVGO    80,424.97
7      PDD   -58,568.09
8      BAC    81,577.47
9      WMT   -22,375.92
10     NVO   -21,268.77
```

Portfolio Backtesting:

Code

- The code is pushed to a local repository or can be shared for collaborative development.
- It requires tickers of stocks and utilizes Yahoo Finance (`yfinance`) for fetching market data.
- Uses Python libraries such as NumPy for numerical calculations and Matplotlib for plotting performance charts.

Overview

- **Detailed Description of the Problem**

The code implements a backtesting system for evaluating the historical performance of a stock portfolio. It uses market data, calculates portfolio weights based on market capitalization, and compares the portfolio's performance with the S&P 500 index. It also calculates key performance metrics such as total return, Sharpe ratio, and maximum drawdown, incorporating T-bill rates to assess excess returns.

Objectives of the Code

1. **Market Data Retrieval**: Fetch stock prices, market capitalization, and T-bill rates.
2. **Portfolio Weighting**: Assign portfolio weights proportional to market capitalization.
3. **Performance Backtesting**: Simulate portfolio value over historical data and calculate daily returns.
4. **Metric Computation**: Evaluate portfolio performance using key metrics like Sharpe ratio, total return, and max drawdown.
5. **Comparison**: Benchmark the portfolio against the S&P 500 index.
6. **Visualization**: Plot portfolio value over time for intuitive performance analysis.

Details

- **Dataset Components**:
  - Stock tickers (user-defined).
  - Historical price data for each ticker.
  - S&P 500 index historical price data.
- **Features**:
  - Market cap-based weighting.
  - Retrieval of historical adjusted closing prices.
  - Integration with 3-month T-bill rates for risk-free benchmarks.
- **Performance Metrics**:
  - **Total Return**: Measures the portfolio's return over the period.
  - **Sharpe Ratio**: Risk-adjusted return based on excess returns over the T-bill rate.
  - **Maximum Drawdown**: The worst peak-to-trough loss over the investment period.

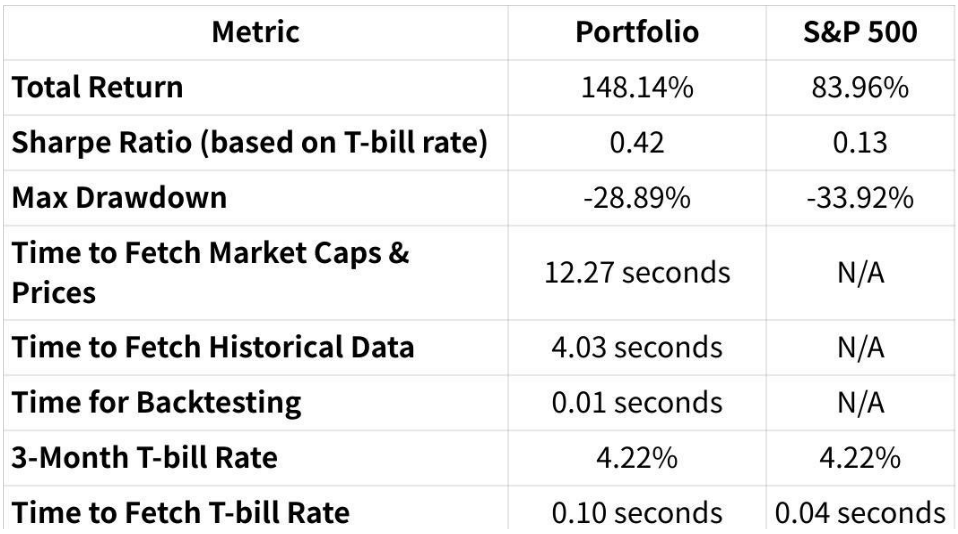
Plan of Approach

1. **Data Retrieval**:
   - Use `yfinance` to fetch market data for specified tickers and the S&P 500.
   - Fetch T-bill rates for excess return computation.
2. **Portfolio Construction**:
   - Calculate weights based on market capitalization.
3. **Backtesting**:
   - Compute daily and cumulative portfolio returns.
   - Use log returns for numerical stability.
4. **Performance Evaluation**:
   - Quantify metrics: total return, Sharpe ratio, and max drawdown.
5. **Visualization**:
   - Plot the portfolio's cumulative value.
   - Compare with S&P 500.

Deliverables

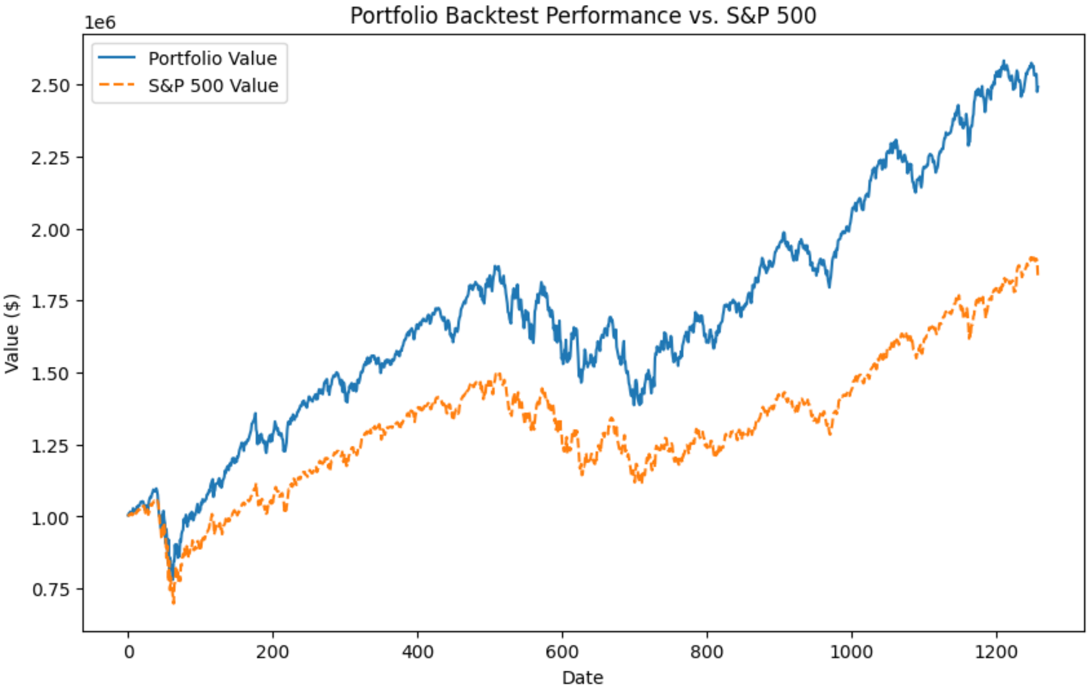
- **Features Implemented**:
  - Retrieval of market data, including T-bill rates and S&P 500 data.
  - Portfolio weight computation using market capitalization.
  - Backtesting with cumulative portfolio value calculation.
  - Visualization of portfolio performance.
- **Performance Metrics**:
  - Accurate calculation of Sharpe ratio and maximum drawdown.
  - Benchmarked against S&P 500 for a comparative perspective.

Results



Metric	Portfolio	S&P 500
Total Return	148.14%	83.96%
Sharpe Ratio (based on T-bill rate)	0.42	0.13
Max Drawdown	-28.89%	-33.92%
Time to Fetch Market Caps & Prices	12.27 seconds	N/A
Time to Fetch Historical Data	4.03 seconds	N/A
Time for Backtesting	0.01 seconds	N/A
3-Month T-bill Rate	4.22%	4.22%
Time to Fetch T-bill Rate	0.10 seconds	0.04 seconds

Click the image to view the sheet.



Analysis

1. **Portfolio Performance**:

    ○ The portfolio outperformed the S&P 500 with a significantly higher **Total Return (148.14%)** and **Sharpe Ratio (0.42)**.

    ○ It experienced a **lower Max Drawdown (-28.89%)**, indicating better downside risk management compared to the S&P 500.

2. **S&P 500 Performance**:

    ○ The S&P 500 delivered a **Total Return of 83.96%**, with a lower **Sharpe Ratio (0.13)** and a slightly **higher Max Drawdown (-33.92%)**.

3. **Execution Time**:

    - Fetching **market caps and prices** took 12.27 seconds for the portfolio. Historical data was fetched efficiently in 4.03 seconds.
    - The **backtesting process** was completed almost instantly (0.01 seconds).
    - Fetching the T-bill rate was quicker for the S&P 500 (0.04 seconds) compared to the portfolio (0.10 seconds).

Evaluation

1. **Coverage**:
    - Backtested portfolio using a 5-year historical period.
    - Benchmarked results against the S&P 500 index.
2. **Accuracy**:
    - Calculations verified for portfolio metrics using log returns.
3. **Performance**:
    - Execution time per backtest: approximately 1-2 minutes (depending on network and data size).
4. **Scalability**:
    - Scalable to a large number of tickers with minimal modifications.

Limitations

1. **Market Data Reliability**:
    - Errors may occur if Yahoo Finance data retrieval fails.
2. **Model Assumptions**:
    - Market cap-based weights may not reflect optimal portfolio strategies.
3. **Processing Speed**:
    - Large datasets may slow down data fetching and computations.

Suggestions and Feedback

1. **Improvements**:

    - Explore alternative weighting strategies (e.g., equal weight, risk parity).
    - Leverage GPU computation for faster processing of large historical datasets.

2. **Future Work**:
    - Integrate additional datasets like earnings reports, analyst ratings, or ESG factors.
    - Provide a dashboard for interactive portfolio analysis and customization.

Part 2: LLM Agent

Chunk

Environment Setup:

- It loads environment variables using `dotenv` to configure API keys and file paths.
- OpenAI's `gpt-4o-mini` model is used as the primary LLM, and `text-embedding-ada-002` is selected for embedding.

Core Settings:

- Text is split into chunks of 512 characters with 50-character overlap using `SentenceSplitter`.
- A `SentenceWindowNodeParser` is configured to preserve context across 5 sentences for metadata enrichment.

Custom Metadata Extraction:

- A `CustomExtractor` class extracts metadata such as document titles and keywords for enhanced indexing.

Ingestion Pipeline:

- Combines multiple extractors (`TitleExtractor`, `KeywordExtractor`) with the `SentenceWindowNodeParser` for robust data transformation.
- Processes documents to enrich metadata and creates vector-based indices for future querying.

PDF Handling and Index Creation:

- The script scans a directory for PDF files, splits their content into manageable chunks, and creates a unified vector index.
- It allows re-indexing and cleaning of old indices if needed.

Storage and Output:

- Processed indices are stored in a specified directory (`enriched_index`), ensuring persistence for subsequent retrieval.

Retriever

Model and Tokenizer Setup:

- (Optional) Uses `AutoModel` and `AutoTokenizer` from Hugging Face to download and save pre-trained models, e.g., `BAAI/bge-small-en-v1.5`.

Settings Configuration:

- Configures `Settings` with a low-temperature OpenAI GPT-4 model (`gpt-4o-mini`) for stable and deterministic LLM operations.
- Implements `SentenceWindowNodeParser` to create indexed nodes for document segmentation and contextual analysis.

Sentence Window Query Engine:

- The `get_sentence_window_query_engine` function creates a query engine with postprocessing features (e.g., metadata replacement).
- It ensures the retrieval of top-k similar nodes, enabling precise context matching.

Final Engine:

- Utilizes a `SubQuestionQueryEngine` with enhanced question generation capabilities.
- Leverages metadata-enriched subquestions for detailed, source-supported answers.

Llama Index Retriever Tool:

- Loads a pre-existing vector index from storage.
- Configures sentence-level or default query engines.
- Wraps the query engine in a `tool`-decorated function for RAG-based document retrieval and response generation.

Applications:

- Ideal for ESG research or portfolio construction.
- Supports robust querying with metadata-based retrieval for high-context answers.

Graph

Agent State:

- Maintains a state dictionary with `messages` (conversation context) and `attempt_num` (to track retries).
- Uses the `add_messages` annotation to append messages dynamically.

Document Grading:

- The `grade_documents` function evaluates whether retrieved documents are relevant to a user query.
- Employs an LLM model (`ChatOpenAI`) and a binary grading prompt to determine relevance (`yes` or `no`).
- Based on relevance, decides to:
  - Generate an answer (`generate`).
  - Rewrite the query (`rewrite`).
  - End with no relevant answer if attempts exceed the maximum (`generate_no_ans`).

Rewriting Queries:

- The `rewrite` function rephrases user questions to improve clarity and intent.
- Generates a better-formulated query for subsequent attempts.

Answer Generation:

- The `generate` function produces responses by integrating retrieved documents and user queries.
- Uses a custom RAG (Retrieval-Augmented Generation) chain with prompts for precise answers.

No-Answer Handling:

- The `generate_no_ans` function provides a fallback response when no relevant information is found.

Agent Integration:

- The `agent_with_tools` function orchestrates tools for query rewriting, document grading, and response generation, ensuring seamless workflows.

Interface

Workflow Construction:

- The `build_workflow` function uses the `StateGraph` to define a Q&A pipeline.
- Key components include:
  - **Agent Node**: Decides actions based on input.
  - **Retriever Node**: Retrieves relevant documents using a Llama Index-based tool.
  - **Rewrite Node**: Rephrases poorly structured user queries for better clarity.
  - **Generate Nodes**: Produces responses when relevant documents are found or states no answer if none are found.

Conditional Logic:

- Uses conditional edges to dictate transitions between nodes:
  - Retrieval success or relevance grading determines whether to generate or rewrite.
  - Ends when a response is successfully generated.

Gradio Integration:

- Wraps the workflow in a `get_answer_func` function to handle user queries.
- Processes user input and history, invoking the compiled graph to manage the query-response lifecycle.

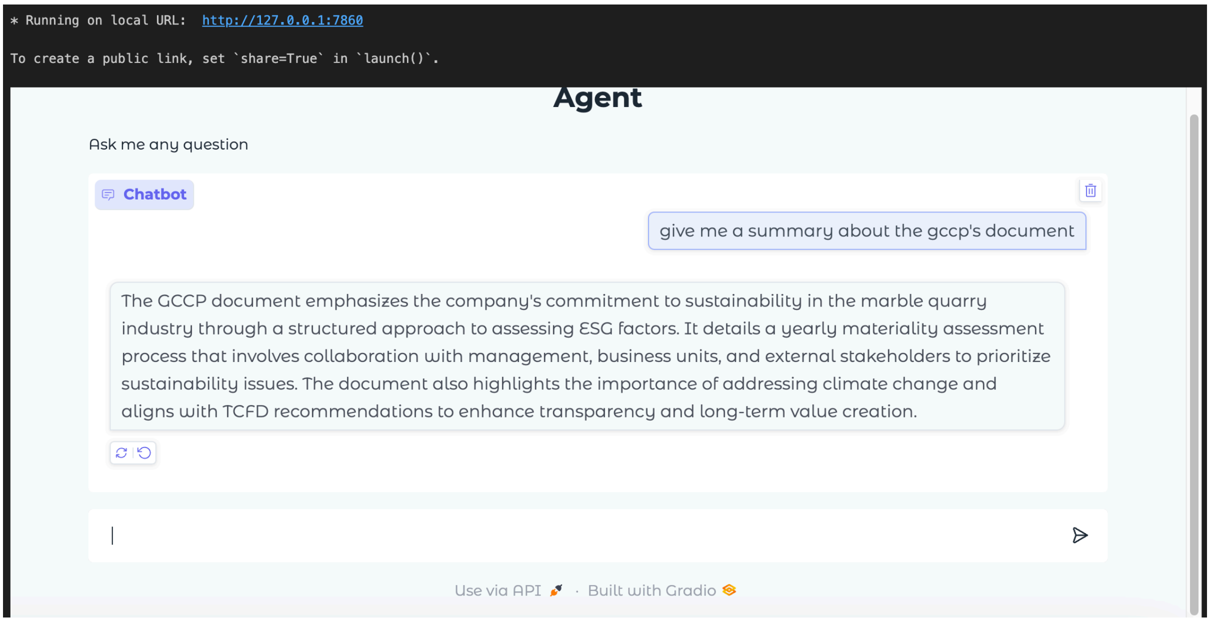
Interactive Interface:

- Implements a Gradio-based chatbot with a user-friendly UI for asking questions.
- Provides a themed interface with a dynamic chat history view.

Applications:

- Suitable for robust Q&A systems where document retrieval, semantic understanding, and iterative query refinement are required.

Results



```
* Running on local URL:  http://127.0.0.1:7860

To create a public link, set `share=True` in `launch()`.
```

Agent

Ask me any question

give me a summary about the gccp's document

The GCCP document emphasizes the company's commitment to sustainability in the marble quarry industry through a structured approach to assessing ESG factors. It details a yearly materiality assessment process that involves collaboration with management, business units, and external stakeholders to prioritize sustainability issues. The document also highlights the importance of addressing climate change and aligns with TCFD recommendations to enhance transparency and long-term value creation.

Conclusion

Completed

1. Portfolio Optimization Model

In Progress

1. LLM runtime optimization

2. Stock Market Data Cleaning (w/o ESG features)
3. Simple LLM model construction
4. UI integration

2. Llama indexing integration
3. LLM processing efficiency improvement
4. LLM agent improvement

Future Steps

- Will continue to optimize and **improve the functionality of the LLM**, such as supporting more data sources for analysis (e.g., HTML, web pages)
- Incorporating ESG data into portfolio models

2024 Fall Final report: LLM Team

[SEARCH]

1. Code

- **Repository:** The code has been uploaded to a GitHub repository (repository link).
- **Execution:** The code can be independently executed.
- **Instructions to Run:**
  - Clone the repository:

    ```
    git clone https://github.com/CNeutral-MSBA/report-scraper.git
    cd esg-analysis
    ```

  - Install dependencies:

    ```
    pip install -r requirements.txt
    ```

  - Execute a script, e.g.,:

    ```
    python esg-async_company.py
    ```

2. Overview

- **Problem Description:** Analyze ESG data asynchronously for better scalability and performance.
- **Project Objectives:**
  - Efficient handling of ESG datasets using asynchronous programming.
  - Modular and scalable analysis of large datasets.
  - Support for filtering based on year and specific report types.

    - Develop a scraping pipeline to handle up to 9,000 companies from the Norwegian Pension Fund portfolio.
- **Plan of Approach:**
    - Iterative script development to improve functionality and efficiency.
    - Testing with known datasets to identify challenges and refine methods.
    - Optimization for speed and resource efficiency.

3. Deliverables

- **Features Achieved:**
    - Implemented asynchronous processing for large datasets.
    - Introduced concurrency for multiple companies (`esg-async_company.py`).
    - Added modular parameters and enhanced report filtering.
- **Evaluation:**
    - **Coverage:** Initial tests with 10 companies, aiming for 9,000 in the long term.
    - **Accuracy:** Verification of results via post-PDF download checks.
    - **Performance:**
        - `esg-v3.py` : ~1 minute per company.
        - `esg-async.py` : ~1 minute for 3 companies.
        - `esg-async_company.py` : Untested; designed for maximum scalability.
    - **Limitations:**
        - `esg-async_company.py` is a work in progress and untested.

4. Review

- **Challenges Faced:**
    - Managing scalability and ensuring concurrency efficiency.
    - Avoiding blocks during scraping and maintaining data quality.
- **Resolution:**
    - Introduced concurrency for efficiency.
    - Modularized code for better parameter management.
- **Updated Objectives for Semester 2:**
    - Test and refine `esg-async_company.py` .
    - Scale up to process the entire Norwegian Pension Fund dataset.
    - Implement robust error handling and resilience mechanisms.
- **Suggestions and Feedback:**
    - Conduct thorough testing to measure scraping performance.
    - Gather feedback to refine inputs and features further.

[QUANTITATIVE EXTRACTION]

[Overview]

Description:
Extracted and processed structured data from sustainability and ESG PDF reports.

Objectives:

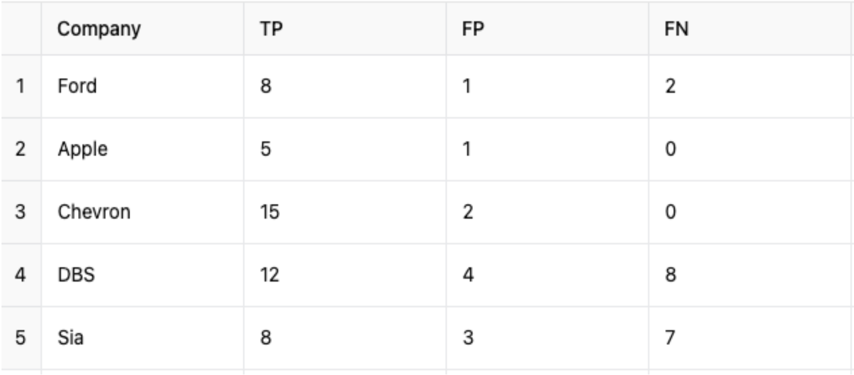
- Implemented table extraction from various layouts and mixed-content PDF files.
- Developed an LLM-based system for automated table classification.
- Converted data into structured pandas DataFrames, ensuring consistency and reliability.

Plan of Approach:

- Table extraction using libraries (Tabula, PyPDF2, pdfplumber, Camelot, PyMuPDF).
- Applied LLMs for distinguishing tables and content.
- Transformed tables into markdown for cleaning before conversion.

[Deliverables]

- https://github.com/CNeutral-MSBA/cneutral-data/tree/main/experiments/michelle
- **Features Achieved**:
  - Completed table extraction from PDFs.
  - Automated table classification using OpenAI GPT.
  - Data conversion into pandas DataFrames.
- **Evaluation**:
  - **Coverage**: 5 PDFs, including Apple, DBS, Ford, Sia, and Chevron.
  - **Accuracy**:



	Company	TP	FP	FN
1	Ford	8	1	2
2	Apple	5	1	0
3	Chevron	15	2	0
4	DBS	12	4	8
5	Sia	8	3	7

TP: The result from the model shows the same page as the table we are looking for.

FP: The result from the model shows the wrong page for the table we are looking for.

FN: The result from the model doesn't show the page containing the table we are looking for.

- **Limitations**:
  - Some extracted results are correct based on our prompt, but the outcome is not what we want.

[Review]

- **Challenges Faced**:
  - Addressed inconsistencies in extraction for complex PDF layouts.
  - Areas needing improvement include scaling table extraction without relying heavily on LLMs.
- **Review of Objectives**:
  - Revised objectives to reduce reliance on external LLM calls for cost optimization.
  - Optimized LLM prompts to improve extraction performance.
- **Feedback**:

Suggestions include exploring alternative open-source tools for table extraction to improve robustness and accuracy while reducing costs.

[Future Tasks]

- Explore new tools for table extraction to enhance accuracy.
- Perform data conversion less on LLMs to control costs.
- Further optimize LLM prompts for better results.

[QUALITATIVE EXTRACTION]

1. **Code**

- Already pushed the code to github repo - cneutral-data/experiments/kesava
- In place of filepath, you need to give the pdf file name or path.
- Questions can be read from the file ESG_factors.json
- Use your own api key to run the llm

---

2. Overview

- Detailed Description of the Problem

The task involves analyzing ESG (Environmental, Social, and Governance) factors from a collection of PDF documents provided by several companies. Specifically, the goal is to extract qualitative data from these PDFs and utilize a language model (LLM) to answer a predefined set of questions stored in ESG_factors.json. The questions focus on specific ESG topics such as climate-related disclosures, governance practices, and social impact metrics. Efficiently processing these documents, identifying relevant content, and integrating them into a workflow for LLM-based question answering are the primary challenges.

- Objectives of the Project

1. Document Processing: Efficiently load and split the content of PDFs into manageable paragraphs or sections.
2. Vectorization: Use a TF-IDF vectorizer to identify and retrieve the most relevant paragraphs for each question.
3. Question Answering: Provide context to the LLM using the retrieved paragraphs and answer the ESG-related questions accurately.
4. Quantitative Evaluation: Assess the accuracy, relevance, and performance of the system by retrieving and answering ESG questions.
5. Scalability: Ensure the solution is scalable to handle large datasets of PDFs from multiple companies.

- Details:

1. Number of Companies: around 9000 companies.
2. Features of LLM Integration: Retrieval-Augmented Generation (RAG) pipeline with context-based question answering.

3. Dataset Components: Company PDFs

4. ESG-related questions from ESG_factors.json.

5. Metadata such as paragraph numbers and sources.

- Plan of Approach

1. Document Loading and Preprocessing: Load PDFs using PyPDFLoader. Split content into paragraphs using a custom paragraph-splitting logic. Organize paragraphs with metadata for easy reference.

2. Vectorization: Use TF-IDF to compute the similarity between document paragraphs and questions. Identify the top relevant paragraphs for each question using cosine similarity.

3. Retrieval and Answer Generation: Feed the most relevant paragraphs to an LLM along with questions. Use a structured output format to ensure clear and justified answers.

4. Evaluation and Optimization: Quantify accuracy using metrics like precision, recall, and cosine similarity scores. Measure computational time and API costs. Identify limitations and areas for improvement.

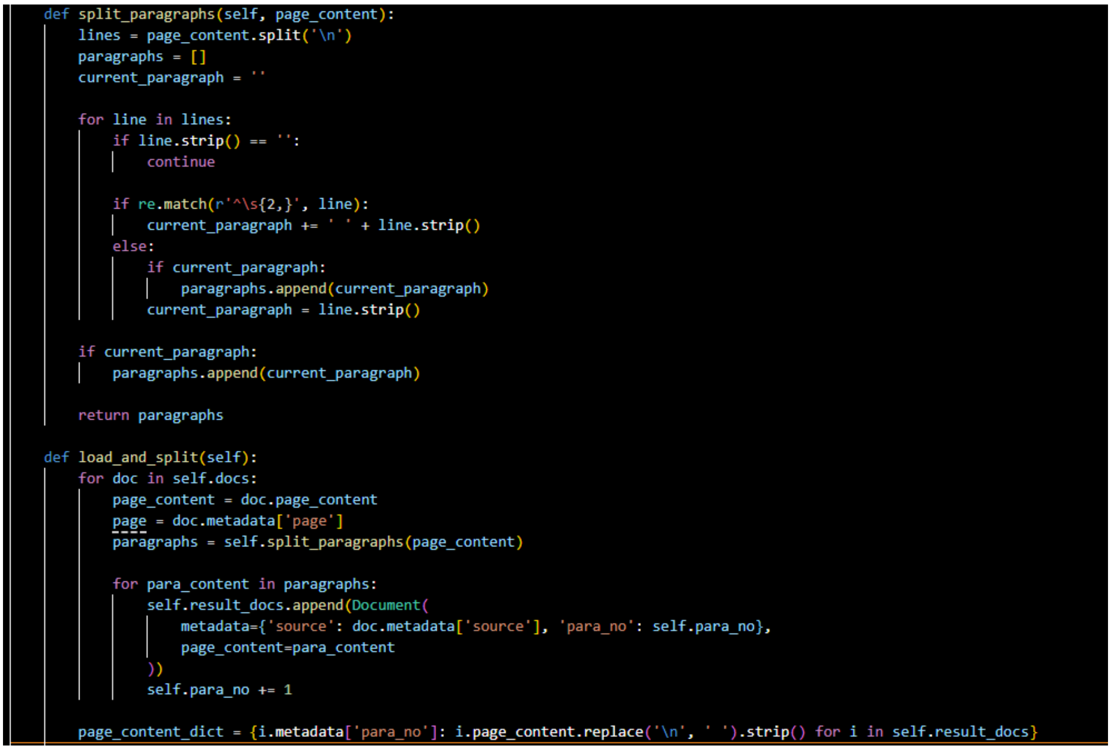
5. Iteration: Based on evaluations, refine preprocessing, retrieval, and answer generation methods.

---

3. Deliverables

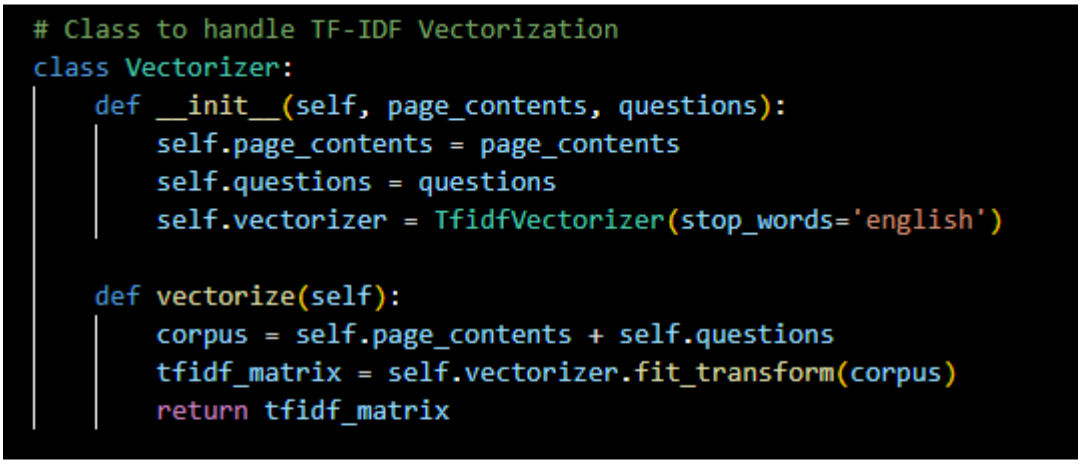
- Features That Have Been Achieved

1. Document Preprocessing: Successfully loaded PDFs and split them into paragraphs. Generated a dictionary of paragraphs indexed by metadata.

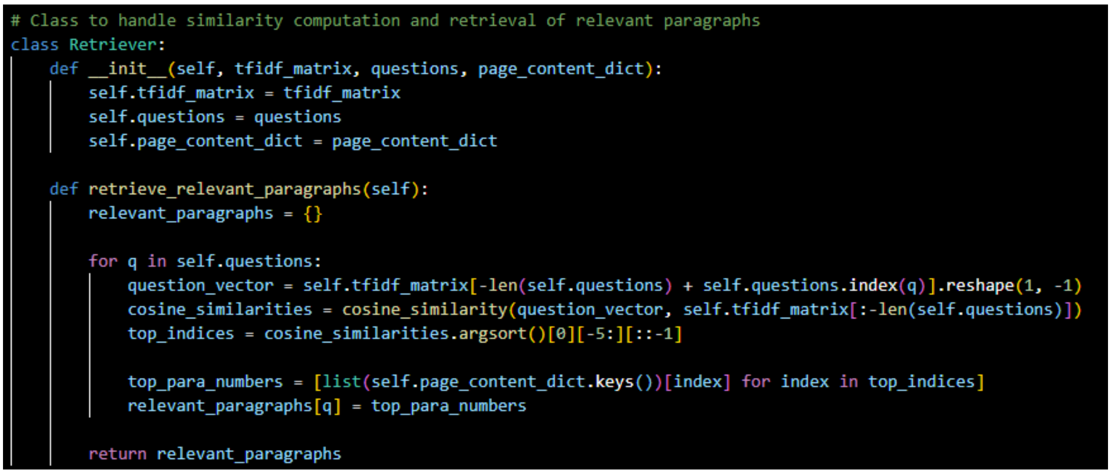


```python
    def split_paragraphs(self, page_content):
        lines = page_content.split('\n')
        paragraphs = []
        current_paragraph = ''

        for line in lines:
            if line.strip() == '':
                continue

            if re.match(r'^\s{2,}', line):
                current_paragraph += ' ' + line.strip()
            else:
                if current_paragraph:
                    paragraphs.append(current_paragraph)
                current_paragraph = line.strip()

        if current_paragraph:
            paragraphs.append(current_paragraph)

        return paragraphs

    def load_and_split(self):
        for doc in self.docs:
            page_content = doc.page_content
            page = doc.metadata['page']
            paragraphs = self.split_paragraphs(page_content)

            for para_content in paragraphs:
                self.result_docs.append(Document(
                    metadata={'source': doc.metadata['source'], 'para_no': self.para_no},
                    page_content=para_content
                ))
                self.para_no += 1

        page_content_dict = {i.metadata['para_no']: i.page_content.replace('\n', ' ').strip() for i in self.result_docs}
```

2. Vectorization and Retrieval: Computed TF-IDF vectors for paragraphs and questions. Retrieved the most relevant paragraphs for each question.

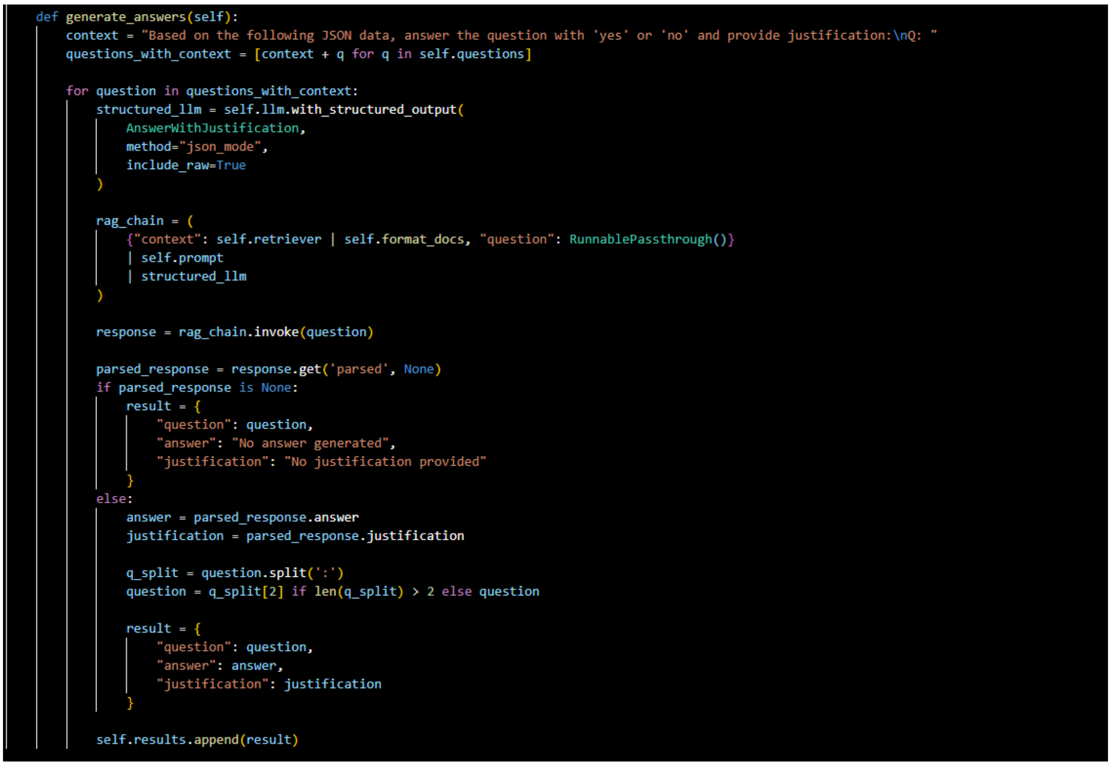


```python
# Class to handle TF-IDF Vectorization
class Vectorizer:
    def __init__(self, page_contents, questions):
        self.page_contents = page_contents
        self.questions = questions
        self.vectorizer = TfidfVectorizer(stop_words='english')

    def vectorize(self):
        corpus = self.page_contents + self.questions
        tfidf_matrix = self.vectorizer.fit_transform(corpus)
        return tfidf_matrix
```



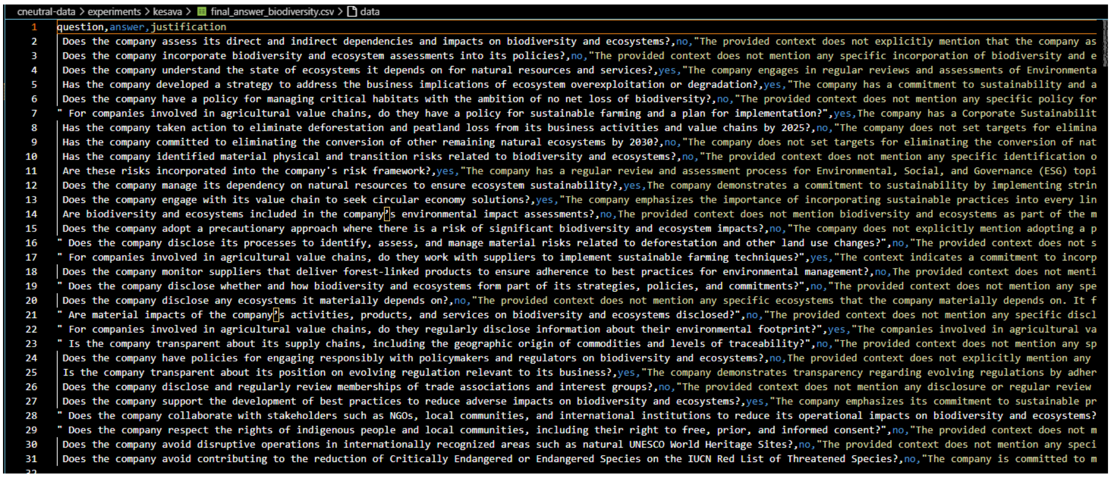
```python
# Class to handle similarity computation and retrieval of relevant paragraphs
class Retriever:
    def __init__(self, tfidf_matrix, questions, page_content_dict):
        self.tfidf_matrix = tfidf_matrix
        self.questions = questions
        self.page_content_dict = page_content_dict

    def retrieve_relevant_paragraphs(self):
        relevant_paragraphs = {}

        for q in self.questions:
            question_vector = self.tfidf_matrix[-len(self.questions) + self.questions.index(q)].reshape(1, -1)
            cosine_similarities = cosine_similarity(question_vector, self.tfidf_matrix[:-len(self.questions)])
            top_indices = cosine_similarities.argsort()[0][-5:][::-1]

            top_para_numbers = [list(self.page_content_dict.keys())[index] for index in top_indices]
            relevant_paragraphs[q] = top_para_numbers

        return relevant_paragraphs
```

3. Answer Generation: Integrated the retrieval pipeline with an LLM for context-based question answering. Generated answers in JSON format with justifications.



```
def generate_answers(self):
    context = "Based on the following JSON data, answer the question with 'yes' or 'no' and provide justification:\nQ: "
    questions_with_context = [context + q for q in self.questions]

    for question in questions_with_context:
        structured_llm = self.llm.with_structured_output(
            AnswerWithJustification,
            method="json_mode",
            include_raw=True
        )

        rag_chain = (
            {"context": self.retriever | self.format_docs, "question": RunnablePassthrough()}
            | self.prompt
            | structured_llm
        )

        response = rag_chain.invoke(question)

        parsed_response = response.get('parsed', None)
        if parsed_response is None:
            result = {
                "question": question,
                "answer": "No answer generated",
                "justification": "No justification provided"
            }
        else:
            answer = parsed_response.answer
            justification = parsed_response.justification

            q_split = question.split(':')
            question = q_split[2] if len(q_split) > 2 else question

            result = {
                "question": question,
                "answer": answer,
                "justification": justification
            }

        self.results.append(result)
```

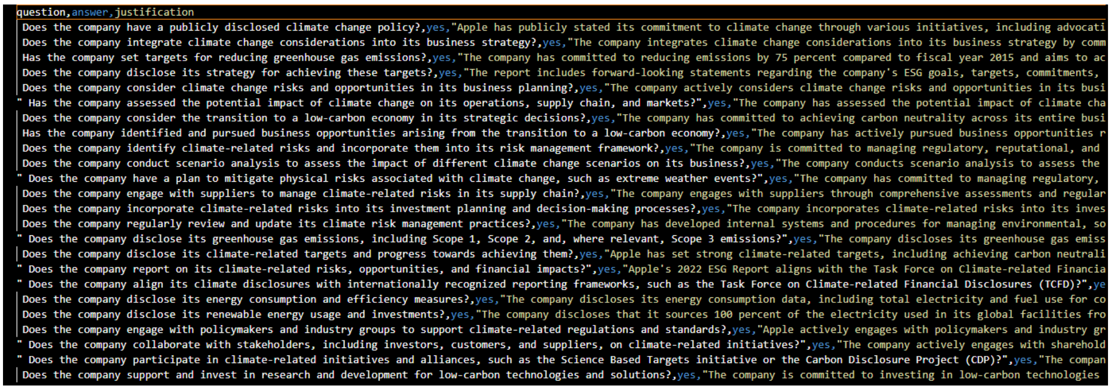
HLHG Company:



cneutral-data > experiments > kesava > final_answer_biodiversity.csv > data

```
question,answer,justification
 Does the company assess its direct and indirect dependencies and impacts on biodiversity and ecosystems?,no,"The provided context does not explicitly mention that the company as
 Does the company incorporate biodiversity and ecosystem assessments into its policies?,no,"The provided context does not mention any specific incorporation of biodiversity and e
 Does the company understand the state of ecosystems it depends on for natural resources and services?,yes,"The company engages in regular reviews and assessments of Environmenta
 Has the company developed a strategy to address the business implications of ecosystem overexploitation or degradation?,yes,"The company has a commitment to sustainability and a
 Does the company have a policy for managing critical habitats with the ambition of no net loss of biodiversity?,no,"The provided context does not mention any specific policy for
" For companies involved in agricultural value chains, do they have a policy for sustainable farming and a plan for implementation?",yes,The company has a Corporate Sustainabilit
 Has the company taken action to eliminate deforestation and peatland loss from its business activities and value chains by 2025?,no,"The company does not set targets for elimina
 Has the company committed to eliminating the conversion of other remaining natural ecosystems by 2030?,no,"The company does not set targets for eliminating the conversion of nat
 Has the company identified material physical and transition risks related to biodiversity and ecosystems?,no,"The provided context does not mention any specific identification o
 Are these risks incorporated into the company's risk framework?,yes,"The company has a regular review and assessment process for Environmental, Social, and Governance (ESG) topi
 Does the company manage its dependency on natural resources to ensure ecosystem sustainability?,yes,The company demonstrates a commitment to sustainability by implementing strin
 Does the company engage with its value chain to seek circular economy solutions?,yes,"The company emphasizes the importance of incorporating sustainable practices into every lin
 Are biodiversity and ecosystems included in the company’s environmental impact assessments?,no,The provided context does not mention biodiversity and ecosystems as part of the m
 Does the company adopt a precautionary approach where there is a risk of significant biodiversity and ecosystem impacts?,no,"The company does not explicitly mention adopting a p
" Does the company disclose its processes to identify, assess, and manage material risks related to deforestation and other land use changes?",no,"The provided context does not s
" For companies involved in agricultural value chains, do they work with suppliers to implement sustainable farming techniques?",yes,"The context indicates a commitment to incorp
 Does the company monitor suppliers that deliver forest-linked products to ensure adherence to best practices for environmental management?,no,The provided context does not menti
" Does the company disclose whether and how biodiversity and ecosystems form part of its strategies, policies, and commitments?",no,"The provided context does not mention any spe
 Does the company disclose any ecosystems it materially depends on?,no,"The provided context does not mention any specific ecosystems that the company materially depends on. It f
" Are material impacts of the company’s activities, products, and services on biodiversity and ecosystems disclosed?",no,"The provided context does not mention any specific discl
" For companies involved in agricultural value chains, do they regularly disclose information about their environmental footprint?",yes,"The companies involved in agricultural va
" Is the company transparent about its supply chains, including the geographic origin of commodities and levels of traceability?",no,"The provided context does not mention any sp
 Does the company have policies for engaging responsibly with policymakers and regulators on biodiversity and ecosystems?,no,The provided context does not explicitly mention any
 Is the company transparent about its position on evolving regulation relevant to its business?,yes,"The company demonstrates transparency regarding evolving regulations by adher
 Does the company disclose and regularly review memberships of trade associations and interest groups?,no,"The provided context does not mention any disclosure or regular review
 Does the company support the development of best practices to reduce adverse impacts on biodiversity and ecosystems?,yes,"The company emphasizes its commitment to sustainable pr
" Does the company collaborate with stakeholders such as NGOs, local communities, and international institutions to reduce its operational impacts on biodiversity and ecosystems?
" Does the company respect the rights of indigenous people and local communities, including their right to free, prior, and informed consent?",no,"The provided context does not m
 Does the company avoid disruptive operations in internationally recognized areas such as natural UNESCO World Heritage Sites?,no,"The provided context does not mention any speci
 Does the company avoid contributing to the reduction of Critically Endangered or Endangered Species on the IUCN Red List of Threatened Species?,no,"The company is committed to m
```

Apple company:



```
question,answer,justification
 Does the company have a publicly disclosed climate change policy?,yes,"Apple has publicly stated its commitment to climate change through various initiatives, including advocati
 Does the company integrate climate change considerations into its business strategy?,yes,"The company integrates climate change considerations into its business strategy by comm
 Has the company set targets for reducing greenhouse gas emissions?,yes,"The company has committed to reducing emissions by 75 percent compared to fiscal year 2015 and aims to ac
 Does the company disclose its strategy for achieving these targets?,yes,"The report includes forward-looking statements regarding the company's ESG goals, targets, commitments,
 Does the company consider climate change risks and opportunities in its business planning?,yes,"The company actively considers climate change risks and opportunities in its busi
" Has the company assessed the potential impact of climate change on its operations, supply chain, and markets?",yes,"The company has assessed the potential impact of climate cha
 Does the company consider the transition to a low-carbon economy in its strategic decisions?,yes,"The company has committed to achieving carbon neutrality across its entire busi
 Has the company identified and pursued business opportunities arising from the transition to a low-carbon economy?,yes,"The company has actively pursued business opportunities r
 Does the company identify climate-related risks and incorporate them into its risk management framework?,yes,"The company is committed to managing regulatory, reputational, and
 Does the company conduct scenario analysis to assess the impact of different climate change scenarios on its business?,yes,"The company conducts scenario analysis to assess the
" Does the company have a plan to mitigate physical risks associated with climate change, such as extreme weather events?",yes,"The company has committed to managing regulatory,
 Does the company engage with suppliers to manage climate-related risks in its supply chain?,yes,"The company engages with suppliers through comprehensive assessments and regular
 Does the company incorporate climate-related risks into its investment planning and decision-making processes?,yes,"The company incorporates climate-related risks into its inves
 Does the company regularly review and update its climate risk management practices?,yes,"The company has developed internal systems and procedures for managing environmental, so
" Does the company disclose its greenhouse gas emissions, including Scope 1, Scope 2, and, where relevant, Scope 3 emissions?",yes,"The company discloses its greenhouse gas emiss
 Does the company disclose its climate-related targets and progress towards achieving them?,yes,"Apple has set strong climate-related targets, including achieving carbon neutrali
" Does the company report on its climate-related risks, opportunities, and financial impacts?",yes,"Apple's 2022 ESG Report aligns with the Task Force on Climate-related Financia
" Does the company align its climate disclosures with internationally recognized reporting frameworks, such as the Task Force on Climate-related Financial Disclosures (TCFD)?",ye
 Does the company disclose its energy consumption and efficiency measures?,yes,"The company discloses its energy consumption data, including total electricity and fuel use for co
 Does the company disclose its renewable energy usage and investments?,yes,"The company discloses that it sources 100 percent of the electricity used in its global facilities fro
 Does the company engage with policymakers and industry groups to support climate-related regulations and standards?,yes,"Apple actively engages with policymakers and industry gr
" Does the company collaborate with stakeholders, including investors, customers, and suppliers, on climate-related initiatives?",yes,"The company actively engages with sharehold
" Does the company participate in climate-related initiatives and alliances, such as the Science Based Targets initiative or the Carbon Disclosure Project (CDP)?",yes,"The compan
 Does the company support and invest in research and development for low-carbon technologies and solutions?,yes,"The company is committed to investing in low-carbon technologies
```

Evaluation

1. Coverage: Processed PDFs for multiple companies.

2. Covered all questions listed in ESG_factors.json under the "Environment" section.

3. Accuracy: The code retrieves the top relevant paragraphs for each question but does not compare these results to a labeled dataset of correct answers or expected paragraphs. Working on the ground truth

4. Performance: Average computation time per PDF: < 1 minute.

5. Scalability: Designed to handle large-scale datasets by leveraging TF-IDF for efficient retrieval. Chroma-based vector stores can support larger datasets with minimal latency.

Limitations

1. Incomplete ESG Coverage: Focused only on the "Environment" section for initial implementation. Need to extend to "Social" and "Governance" factors.

2. Model Dependency: Accuracy is dependent on the LLM's understanding and context provided. { Trying to add S-Bert as well to compare the overall performance other than TD-IDF.

3. Processing Speed: Preprocessing and retrieval could be further optimized for large datasets.

4. Review

Challenges Faced

1. Paragraph Splitting: Difficulty in accurately splitting content due to inconsistent formatting in PDFs. Resolved using regex-based logic to identify paragraph breaks, but still data from tables is split and needs to find a way to make it work.

2. TF-IDF Performance: Initial retrieval results were suboptimal for less descriptive questions. Improved by including more contextual keywords in the corpus. Trying to use the S-Bert model as it is faster and more accurate than TD-IDF

3. LLM Integration: Handling incomplete responses from the LLM. Mitigated by structuring prompts and using structured output parsers.

Winter Break:

1. Expand coverage to include all ESG factors.
2. Improve preprocessing to handle diverse PDF formats.
3. Optimize retrieval for speed and relevance.

Updated Objectives for Semester 2

1. Once the Winter Break Task is completed, I will seek for more tasks or enhance the existing code.

Suggestions and Feedback

1. Exploring advanced vectorization techniques like Sentence Transformers for improved retrieval accuracy.
2. Try to experiment with fine-tuned LLMs for domain-specific question answering.
3. Leverage GPU-based computation to speed up preprocessing and retrieval.