Full stack data scientist hiring assignment

scRNA seq data ingestion & visualization

Problem 1: Ingesting and Labeling Public scRNA-seq Datasets

You are required to download two publicly available scRNA-seq datasets:

- **Diseased mice dataset** from GSE180498, including samples with the following IDs:
  - GSM5463957
  - GSM5463958
  - GSM5463959
- **Control mice dataset** from GSE142541, including samples with the following IDs:
  - GSM4231663
  - GSM4231664
  - GSM4231665
  - GSM4231666
  - GSM4231667
  - GSM4231668

Tasks:

1. Ingest both datasets into `AnnData` format using `scanpy`.
2. Label the samples by adding two columns to `adata.obs`:
   - **`sample`**: Assign a unique label to each sample (e.g., `NODAIRE_1`, `NODAIRE_2`, `NODCTRL_1`, etc.).
   - **`cohort`**: Indicate whether the sample belongs to the NODAIRE, SC or NODICAM group.

Problem 2: Visual Inspection and Quality Control (QC) of scRNA-seq Data

Before applying quality control (QC) to remove bad quality cells and genes, you must first conduct a visual inspection of the data to inform your rationale for selecting cutoffs. You will also need to identify potential doublets.

Tasks:

1. **Calculate QC Metrics**:
   - For each cell in your dataset, compute the following QC metrics:
     - **Total Counts**: Sum of all counts (RNA content) per cell.
     - **Number of Genes Detected**: Count of unique genes with non-zero expression per cell.
     - **Percentage of Mitochondrial Genes**: Calculate the percentage of mitochondrial counts relative to total counts.

2. **Visual Inspection**:
    - Create the following visualizations to assess the quality of the cells:
        - Distribution of total counts per cell (total RNA content per cell).
        - Number of genes detected per cell.
        - Percentage of mitochondrial genes per cell.
    - Use these plots to guide the selection of thresholds for filtering out low-quality cells, overrepresented cells and cells with high mitochondrial content

---

Problem 3: Code Exploration and Statistical Analysis of `sc.tl.rank_gene_groups`

You are Required to analyze the implementation of the `sc.tl.rank_gene_groups` function in the Scanpy library. This function is essential for identifying differentially expressed genes across specified groups in scRNA seq data. Your task is to read through the code, understand its functionality, and answer the following questions regarding the statistical calculations and methodology.

Task:

1. **Code Exploration**:
    - Locate the implementation of `sc.tl.rank_gene_groups` in the Scanpy GitHub repository or your local installation.
    - Carefully review the code to understand how it processes input data and computes the relevant statistics for gene ranking.
2. **Answer the Following Questions**:
    - **Statistical Tests**: What statistical tests are used within `sc.tl.rank_gene_groups` to evaluate differential expression? Identify the specific lines of code responsible for invoking these tests.
    - **Log2 Fold Change (log2FC)**: Describe the mathematical formula used to compute log2FC for genes between the specified groups. Which lines in the code perform this calculation?
    - **P-values**: Explain how p-values are calculated within the function. What assumptions does the function make about the data (e.g., distribution)? Highlight the relevant lines of code.
    - **Adjustment for Multiple Testing**: How does the function handle multiple testing correction? Specify which method is used and the corresponding lines in the code.
    - **Group Comparison**: Describe how the function distinguishes between conditions when ranking gene groups. What parameters or data structures facilitate this?