Package ‘TCGAimmunosurv’

February 18, 2025

Title Mutation-specific Survival Analysis and Immune Cell Transitions

Version 1.0

Maintainer Devvrat Pandey <pandey.devvrat@thsti.res.in>

Description
‘TCGAimmunosurv’, for the integrated analysis of bulk and single-cell RNA-Seq data across pan-cancer studies. This package facilitates a deeper understanding of cancer immune dynamics in the context of specific oncogene mutations. Key features of this package include mutation-specific survival analysis, pseudotime trajectory analysis, identification of differentially expressed genes, and comprehensive insights into mutations within cancers or specific genes.

License GPL-3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Depends R(>= 4.4.0), maftools, TimeSeriesExperiment

Imports Seurat, SeuratWrappers, SummarizedExperiment, DESeq2,
survival, survminer, ggplot2, dplyr, data.table, stringr,
utils, monocle3, TCGAbiolinks, kableExtra, Matrix, gridExtra,
stats, methods

Remotes satijalab/seurat-wrappers, cole-trapnell-lab/monocle3,
bioinformaticsFMRP/TCGAbiolinks, satijalab/seurat,
PoisonAlien/maftools, thelovelab/DESeq2, Bioconductor/Rhtslib,
nlhuong/TimeSeriesExperiment

Suggests BiocManager, knitr, rmarkdown

NeedsCompilation no

Author Devvrat Pandey [aut, cre],
Shivam Kumar [ctb, aut],
Dipanka Tanu-Sarmah [ctb, aut],
Samrat chatterjee [aut, ctb]

Contents

`assign_cd8_labels` *Assign CD8+ T cell Labels*

Description

This function assigns labels to CD8+ T cells based on marker gene expression (e.g., naive, cytotoxic, exhausted).

Usage

```
assign_cd8_labels(
  seurat_obj,
  all_markers,
  naive_markers,
  cytotoxic_markers,
  exhausted_markers
)
```

Arguments

`seurat_obj` A Seurat object with CD8+ T cell expression data.

`all_markers` A data frame of all markers identified across clusters.

`naive_markers` A vector of naive CD8+ T cell markers.

`cytotoxic_markers`
A vector of cytotoxic CD8+ T cell markers.

`exhausted_markers`
A vector of exhausted CD8+ T cell markers.

Value

A list containing the updated Seurat object with CD8+ labels and a subset of CD8+ T cells.

`calculate_DEGs` *Perform Differential Expression Analysis*

Description

This function calculates differentially expressed genes (DEGs) from RNA-Seq data using the DESeq2 package. It applies thresholds for statistical significance (p-value) and fold change to classify genes as upregulated, downregulated, or not significant.

Usage

```
calculate_DEGs(rna_data, condition_column, alpha, log2FC_threshold)
```

Arguments

`rna_data` A `DESeqDataSet` object containing the RNA-Seq count matrix and metadata.

`condition_column`
A string specifying the name of the column in the metadata that defines the experimental conditions/groups.

`alpha` A numeric value specifying the significance threshold for the p-value (e.g., 0.05).

`log2FC_threshold`
A numeric value specifying the log2 fold change threshold to determine upregulated or downregulated genes.

Value

A data frame containing the DESeq2 results with an additional column, `significance`, indicating whether each gene is upregulated, downregulated, or not significant.

`cluster_and_umap` *Cluster and Perform UMAP*

Description

This function clusters the cells and performs UMAP dimensionality reduction for visualization.

Usage

```
cluster_and_umap(seurat_obj, dims = 20, resolution = 0.1)
```

Arguments

`seurat_obj`	A Seurat object containing single-cell RNA-seq data.
`dims`	Number of dimensions to use for clustering (default is 20).
`resolution`	Resolution parameter for clustering (default is 0.1).

Value

A Seurat object with clustering and UMAP results added.

`extract_genes_hr_gt1` *Extract Significant Genes with Hazard Ratio Greater Than 1 and Save to CSV*

Description

This function filters significant genes with a hazard ratio (HR) greater than 1 from a given results DataFrame. It returns a DataFrame of the filtered genes and their hazard ratios and saves the results to a `.csv` file.

Usage

```
extract_genes_hr_gt1(results, output_file = "significant_genes.csv")
```

Arguments

`results`	A DataFrame containing gene analysis results with at least the following columns: `significant`, `hazard_ratio`, and `gene_name`.
`output_file`	Character. The name of the output `.csv` file to save the results. Default is "significant_genes.csv".

Value

A DataFrame with two columns:

- `gene_name`: Names of significant genes with HR > 1.
- `hazard_ratio`: The hazard ratios of the significant genes.

`filter_rna_data` *Filter RNA Data*

Description

This function filters RNA-Seq data stored in a `SummarizedExperiment` object. It allows for filtering samples based on specific conditions, removing rows with high NA values, replacing remaining NAs with the median, and removing low-count rows.

Usage

```
filter_rna_data(
  rna_data,
  condition_column,
  condition_levels,
  na_threshold = 0.8
)
```

Arguments

`rna_data` A `SummarizedExperiment` object containing RNA-Seq count data.

`condition_column`
A character string specifying the column name in `colData` that contains the conditions for filtering.

`condition_levels`
A character vector of condition levels to retain in the data.

`na_threshold` A numeric value (default = 0.8) specifying the maximum allowed fraction of NAs in a row. Rows with a higher fraction of NAs will be removed.

Value

A filtered `SummarizedExperiment` object with updated counts, `rowRanges`, and `colData`.

`filter_seurat_object` *Filter Seurat Object Based on Quality Control Criteria*

Description

This function filters a Seurat object based on the number of detected features (genes) and the percentage of mitochondrial genes. It ensures that only high-quality cells are kept for downstream analysis.

Usage

```
filter_seurat_object(
  seurat_obj,
  min_features = 200,
  max_features = 2500,
  max_percent_mt = 5
)
```

Arguments

`seurat_obj` A Seurat object containing single-cell RNA-seq data.

`min_features` An integer specifying the minimum number of detected features (genes) required for a cell to be retained. Default is 200.

`max_features` An integer specifying the maximum number of detected features (genes) allowed for a cell. Default is 2500.

`max_percent_mt` A numeric value specifying the maximum percentage of mitochondrial genes allowed for a cell. Default is 5.

Details

The function adds a new metadata column to the Seurat object, `percent.mt`, which represents the percentage of mitochondrial genes expressed in each cell. The cells are then filtered based on the following criteria:

- The number of detected features (genes) must be between `min_features` and `max_features`.
- The percentage of mitochondrial gene expression must be below `max_percent_mt`.

This helps to remove cells with poor quality (e.g., low gene count or high mitochondrial gene expression) that could skew downstream analyses.

Value

A filtered Seurat object, where cells meeting the quality control criteria are retained.

`generate_mutation_summary` *Generate Mutation Summary for Genes of Interest*

Description

This function generates a summary of mutation types and their counts for a given list of genes from a MAF (Mutation Annotation Format) dataset.

Usage

```
generate_mutation_summary(cancer_maf, genes_of_interest)
```

Arguments

`cancer_maf` A MAF object created using the `maftools` package, containing mutation data.

`genes_of_interest` A character vector of gene names to summarize mutations for.

Details

- The function checks if each gene in `genes_of_interest` exists in the MAF data.
- If a gene is not found or has no non-synonymous variants, a warning is issued, and the gene is skipped.
- Mutation types and counts are extracted for each valid gene.

Value

A data table containing the mutation summary for the specified genes. The table includes:

- `Gene`: The gene name.
- `Mutation_Type`: The type of mutation.
- `NumSamples`: The number of samples with the specified mutation type.

`get_mutated_samples` *Get Unique Mutated Sample IDs for Selected Genes*

Description

This function extracts unique tumor sample barcodes for selected genes with mutations from a MAF (Mutation Annotation Format) dataset.

Usage

```
get_mutated_samples(cancer_maf, genes_of_interest)
```

Arguments

`cancer_maf` A MAF object created using the `maftools` package, containing mutation data.

`genes_of_interest` A character vector of gene names for which the mutated samples are to be extracted.

Details

- The function subsets the MAF data to include only the mutations for the specified genes.
- It then extracts the unique tumor sample barcodes where mutations for those genes are present.

Value

A character vector containing the unique tumor sample barcodes of samples where mutations in the selected genes are present.

get_top_combined_genes

Get Top Combined Ranked Genes and Save to CSV

Description

This function identifies the top genes based on a combined rank of q-values and Moran's I statistic from spatial transcriptomics analysis, and saves the results to a .csv file.

Usage

```
get_top_combined_genes(
  graph_test_res,
  q_value_threshold = 0.05,
  top_n = 10,
  output_file = "top_genes.csv"
)
```

Arguments

graph_test_res	A data frame containing gene statistics, including q_value (adjusted p-value) and morans_I (Moran's I statistic).
q_value_threshold	Numeric. The threshold for significant q-values. Default is 0.05.
top_n	Integer. The number of top genes to return based on the combined rank. Default is 10.
output_file	Character. The name of the output .csv file to save the results. Default is "top_genes.csv".

Details

The function filters genes based on the q_value_threshold, then combines the ranks of q_value (ascending order) and morans_I (descending order). The combined rank is used to determine the top top_n genes. The top genes and their ranks are saved to a .csv file.

Value

A character vector containing the names of the top-ranked genes.

`identify_markers` *Identify Cluster Markers*

Description

This function identifies marker genes for each cluster.

Usage

```
identify_markers(seurat_obj, min_pct = 0.25, logfc_thresh = 0.25)
```

Arguments

`seurat_obj` A Seurat object with clustered cells.

`min_pct` Minimum percentage of cells expressing the gene to be considered as a marker (default is 0.25).

`logfc_thresh` Minimum log fold change for a gene to be considered significant (default is 0.25).

Value

A data frame containing identified markers for each cluster.

`load_cancer_data` *Load Cancer Data for a Selected Cancer Type*

Description

This function loads TCGA data for a specified cancer type, including RNA-Seq counts, clinical data, and mutation data. If the data file exists, it is loaded directly.

Usage

```
load_cancer_data(selected_cancer_type)
```

Arguments

`selected_cancer_type` A character string specifying the TCGA cancer type (e.g., "BRCA" for breast cancer, "LUAD" for lung adenocarcinoma).

Details

If the data file exists locally, this function will:

- Load the RNA-Seq count data.
- Load the clinical data.
- Load the mutation data.

Value

None. This function loads the data into the global environment:

- `tcga_count_data`: RNA-Seq count data in a `SummarizedExperiment` object.
- `clinical_data`: Clinical data for survival analysis and metadata.
- `mutation_data`: Mutation data from TCGA.
- `sample_type`: A data frame of sample types.

`load_seurat_object` *Load Seurat Object for Single-Cell Analysis*

Description

This function loads a Seurat object from a specified directory based on a given cancer type. The Seurat object will be used to perform further single-cell RNA-seq analysis.

Usage

```
load_seurat_object(
  directory,
  constant_prefix = "combined_seurat_",
  selected_cancer
)
```

Arguments

`directory` A character string specifying the directory where the Seurat object file is located.

`constant_prefix`
A character string specifying the constant prefix for the Seurat object file name. The default is "combined_seurat_".

`selected_cancer`
A character string specifying the cancer type for which the Seurat object file will be loaded. The cancer type will be appended to the constant prefix.

Details

This function constructs the file name for the Seurat object based on the provided `constant_prefix` and `selected_cancer`. It checks if the file exists in the specified directory and loads the Seurat object if the file is found. If the file does not exist, an error message is displayed.

Value

A Seurat object that has been loaded from the specified file.

`mapping_genes` *Map Differentially Expressed Genes (DEGs) with Gene Metadata*

Description

This function takes in the results of differential gene expression analysis (`deg_results`), a gene metadata table (`gene_metadata_dt`), and filters for upregulated and downregulated genes based on user-defined thresholds for log fold change and adjusted p-value. It then returns a merged table with gene names and the associated differential expression data.

Usage

```
mapping_genes(deg_results, gene_metadata_dt, logFC_threshold, padj_threshold)
```

Arguments

`deg_results` A data frame containing differential expression results. The data frame should include at least the following columns:

- **log2FoldChange** The log2 fold change values for the genes.
- **padj** The adjusted p-values for the genes.
- **ensemble_id** A unique identifier for each gene (can be rownames if not present as a column).

`gene_metadata_dt` A data frame containing gene metadata. This should include at least the following columns:

- **ensemble_id** The unique identifier for the gene, which matches the `ensemble_id` in `deg_results`.
- **gene_name** The gene names corresponding to the `ensemble_id`.

`logFC_threshold` A numeric value specifying the threshold for log2 fold change. Genes with `log2FoldChange` > `logFC_threshold` are considered upregulated, and genes with `log2FoldChange` < `-logFC_threshold` are considered downregulated.

`padj_threshold` A numeric value specifying the adjusted p-value threshold. Only genes with `padj` < `padj_threshold` will be selected.

Value

A data frame containing the upregulated and downregulated genes based on the given thresholds, with columns:

- **ensemble_id** The unique identifier for the gene.
- **gene_name** The gene name associated with the `ensemble_id`.
- **log2FoldChange** The log2 fold change values for the gene.
- **padj** The adjusted p-values for the gene.

perform_pca *Perform Principal Component Analysis (PCA) on Seurat Object*

Description

This function performs PCA on a Seurat object after normalizing the data, identifying variable features, and scaling the data. The resulting PCA components can be used for dimensionality reduction and visualization in downstream analysis.

Usage

```
perform_pca(seurat_obj)
```

Arguments

`seurat_obj`	A Seurat object containing single-cell RNA-seq data.

Details

The function performs the following steps:

- Normalizes the data using `Seurat::NormalizeData()`.
- Identifies highly variable features with `Seurat::FindVariableFeatures()`.
- Scales the data using `Seurat::ScaleData()`.
- Runs PCA using the identified variable features with `Seurat::RunPCA()`.

These steps are essential for reducing the dimensionality of the data and identifying principal components that explain most of the variance in the dataset.

Value

A Seurat object with PCA results stored in the `pca` assay, ready for further analysis (e.g., visualization, clustering).

perform_pseudotime *Perform Pseudotime Analysis*

Description

This function performs pseudotime analysis on CD8+ T cells using the Monocle3 package.

Usage

```
perform_pseudotime(seurat_obj, num_dim = 100, root_cluster = "Naive CD8+ T")
```

Arguments

`seurat_obj` A Seurat object with CD8+ T cells.

`num_dim` Number of dimensions for preprocessing (default is 100).

`root_cluster` The cluster to set as the root (default is "Naive CD8+ T").

Value

A Monocle3 CellDataSet with pseudotime trajectory learned.

`perform_survival_analysis` *Perform Survival Analysis on RNA-Seq Data*

Description

This function performs survival analysis on RNA-Seq data by grouping samples into two strata (HIGH/LOW) based on the median expression of each gene. It then performs Cox regression analysis for each gene and returns the hazard ratios and p-values.

Usage

```
perform_survival_analysis(
  count_matrix_filtered,
  gene_metadata_dt,
  clinical_data_filtered
)
```

Arguments

`count_matrix_filtered`
A matrix of RNA-Seq count data, where rows represent genes and columns represent samples.

`gene_metadata_dt`
A data frame containing metadata for the genes, including a `gene_name` column.

`clinical_data_filtered`
A data frame containing clinical data, including `submitter_id`, `overall_survival`, and `deceased` columns.

Value

A data frame with the results of the survival analysis for each gene, including:

- `gene_name`: The name of the gene.
- `hazard_ratio`: The hazard ratio for the gene.
- `pvalue`: The p-value from the Cox regression for the gene.
- `significant`: Whether the gene is significantly associated with survival (p-value \leq 0.05).

`plot_cd8_trajectory` *Visualize Pseudotime Trajectory*

Description

This function visualizes the pseudotime trajectory for CD8+ T cells.

Usage

```
plot_cd8_trajectory(
  cds,
  cd8_tcells,
  color_by_pseudotime = TRUE,
  label_size = 6
)
```

Arguments

`cds`	A Monocle3 CellDataSet containing pseudotime data.
`cd8_tcells`	A Seurat object containing CD8+ T cell clusters.
`color_by_pseudotime`	A logical value to color by pseudotime (default is TRUE).
`label_size`	Size of labels in the plot (default is 6).

Value

Pseudotime trajectory plot.

`plot_DEG_results` *Plot Differentially Expressed Genes (DEGs) Results*

Description

This function generates two plots to visualize the results of differential expression analysis: a bar plot showing the sample distribution for different conditions and a volcano plot for visualizing the differentially expressed genes (DEGs).

Usage

```
plot_DEG_results(
  res_df,
  rna_data,
  condition_column,
  log2FC_threshold = 1,
  alpha = 0.05
)
```

Arguments

`res_df` A data frame containing the results of the differential expression analysis, including columns for `log2FoldChange`, `padj` (adjusted p-values), and `significance`. The `significance` column should indicate whether a gene is "Upregulated", "Downregulated", or "Not Significant".

`rna_data` A `SummarizedExperiment` object containing RNA-Seq data with sample metadata accessible via `SummarizedExperiment::colData`.

`condition_column`
A character string specifying the name of the column in `SummarizedExperiment::colData(rna_data)` that contains condition labels (e.g., "Control" and "Treatment").

`log2FC_threshold`
A numeric value specifying the threshold for `log2FoldChange`. Default is 1.

`alpha` A numeric value specifying the significance threshold for adjusted p-values. Default is 0.05.

Details

- The bar plot displays the sample count for each condition in the dataset, with labels for the number of samples above each bar.
- The volcano plot uses `log2FoldChange` and `-log10(padj)` to visualize the significance and magnitude of gene expression changes. Points are colored based on their significance status.
- Both plots use customizable font styles for better readability.

Value

None. This function outputs two plots:

- A bar plot showing the distribution of samples across conditions.
- A volcano plot visualizing the DEGs.

`plot_km_curves` *Plot Kaplan-Meier Curves for Significant Genes*

Description

This function generates Kaplan-Meier survival curves for each significant gene based on its association with the clinical data. The function filters the `strata_data` for the significant genes, merges it with clinical data, and fits the Kaplan-Meier model for each gene. A survival plot is then created for each gene showing survival probability over time.

Usage

```
plot_km_curves(strata_data, clinical_data_filtered, significant_genes)
```

Arguments

`strata_data` A data frame containing gene expression data for multiple cases. This data frame should include at least the columns:

gene_name The name of the gene.

case_id The identifier for the case/sample.

`clinical_data_filtered`
A data frame containing clinical data filtered for the relevant cases. This data frame should include at least the columns:

submitter_id The identifier for the case/sample.

overall_survival The survival time of the patient (in days).

deceased A binary indicator of whether the patient is deceased (1 = deceased, 0 = alive).

strata A factor or categorical variable used to stratify the survival analysis (e.g., treatment groups).

`significant_genes`
A character vector containing the names of significant genes to plot. The function will generate Kaplan-Meier curves for each gene in this vector.

Value

A list of Kaplan-Meier plots (`ggsurvplot` objects) for each significant gene. The plots are printed individually for each gene.

`plot_mutation_counts` *Plot Mutation Counts for Each Gene*

Description

This function generates separate bar plots showing the mutation counts for each gene, based on mutation types and sample counts.

Usage

```
plot_mutation_counts(mutation_counts)
```

Arguments

`mutation_counts`
A data.frame or data.table containing the mutation data, including columns for:

- `Gene`: The gene name.
- `Mutation_Type`: The type of mutation (e.g., missense, nonsense).
- `NumSamples`: The number of samples with that mutation type for the given gene.

Details

The function iterates over each unique gene in the `mutation_counts` data and creates a bar plot for that gene. Each plot represents the count of mutations per mutation type for the gene, with the bars filled according to the mutation type.

Value

A series of bar plots, one for each gene, showing the count of mutations by mutation type.

`plot_results` *Plot Survival Analysis Results*

Description

This function generates three types of plots to visualize the survival analysis results: a bar plot showing the number of significant vs. non-significant genes, a bar plot showing the count of significant genes categorized by hazard ratio, and a volcano plot representing the relationship between hazard ratio and p-value.

Usage

```
plot_results(results)
```

Arguments

`results` A data frame containing the results of the survival analysis. The data frame should have the following columns:

- **gene_name** Character string, the name of the gene.
- **hazard_ratio** Numerical value representing the hazard ratio.
- **pvalue** Numerical value representing the p-value of the gene.
- **significant** Character string, "Yes" if the gene is statistically significant, otherwise "No".

Value

A list containing three ggplot objects:

SignificancePlot A bar plot showing the count of significant vs. non-significant genes.

HRPlot A bar plot showing the count of significant genes categorized by hazard ratio.

VolcanoPlot A volcano plot showing the relationship between log hazard ratio and -log10 p-value.

`plot_unique_mutation_types` *Plot Unique Mutation Types Across Genes*

Description

This function generates a bar plot to visualize the distribution of unique mutation types across genes for a given cancer type based on the MAF (Mutation Annotation Format) data.

Usage

```
plot_unique_mutation_types(cancer_maf)
```

Arguments

`cancer_maf`	A `MAF` object representing mutation data for a specific cancer type. This object is typically created using the `maftools::read.maf` function.

Details

- The function extracts the mutation type counts from the `MAF` object using the `maftools::getGeneSummary` function.
- Mutation types with zero counts are filtered out.
- The mutation counts are aggregated across all genes for each mutation type.
- The resulting plot displays mutation types on the x-axis and the total number of samples on the y-axis.

Value

A ggplot object showing the bar plot of unique mutation types and their total number of samples across all genes in the provided `MAF` object.

`preprocess_data` *Preprocess RNA-Seq and Clinical Data*

Description

This function preprocesses RNA-Seq count data and clinical data, filtering both to retain only the samples that have mutation data and are present in both the RNA-Seq and clinical datasets. It also standardizes the sample IDs to a common format.

Usage

```
preprocess_data(rna_data, clinical_data, extracted_mutated_sample_ids)
```

Arguments

`rna_data`	A `SummarizedExperiment` or similar object containing RNA-Seq count data. It must have a count matrix and associated gene metadata.
`clinical_data`	A data frame containing clinical data with a column `submitter_id` that matches sample IDs in `rna_data`.
`extracted_mutated_sample_ids`	A character vector of sample IDs that have mutations of interest.

Details

The function filters the RNA-Seq count matrix and the clinical data to include only the samples that are present in both the mutation data and clinical data. The sample IDs are standardized to the first three components (using hyphen delimiters) for consistency.

Value

A list containing two elements:

- `count_matrix_filtered`: A matrix of RNA-Seq count data for samples that have mutation data and are present in both datasets.
- `clinical_data_filtered`: A data frame of clinical data for samples that have mutation data and are present in both datasets.

`run_graph_test` *Run Graph Test*

Description

This function performs a graph test on the Monocle3 CellDataSet.

Usage

```
run_graph_test(cds, neighbor_graph = "knn", cores = 1)
```

Arguments

`cds`	A Monocle3 CellDataSet to run the graph test on.
`neighbor_graph`	The neighbor graph type to use ("knn" or "principal_graph").
`cores`	Number of CPU cores to use for parallel processing.

Value

Data frame with graph test results.

`standardize_tcga_ids` *Standardize TCGA Sample IDs*

Description

This function standardizes TCGA sample IDs by extracting the first three components of the ID before the first hyphen ("-"). If the ID format is invalid (does not contain "-"), it returns `NA`.

Usage

```
standardize_tcga_ids(sample_ids)
```

Arguments

`sample_ids`	A character vector of TCGA sample IDs.

Details

The function ensures that all TCGA sample IDs are standardized to the format containing only the first three components separated by hyphens. If the input contains IDs in an incorrect format (i.e., not containing "-"), the function returns `NA` for those IDs.

Value

A character vector of standardized TCGA sample IDs, where each ID is truncated to the first three components, or `NA` if the format is invalid.

`style_mutation_table` *Style the Mutation Counts Table*

Description

This function applies styling to a mutation counts table, enhancing the readability and presentation of the data using the `kable` and `kableExtra` packages.

Usage

```
style_mutation_table(mutation_counts)
```

Arguments

`mutation_counts`

A data frame or data table containing the mutation counts, with columns for:

- `Gene`: The gene name.
- `Mutation_Type`: The type of mutation.
- `NumSamples`: The number of samples with that mutation type for the given gene.

Details

The function uses `kableExtra::kbl()` to create a table and then applies various styling options:

- `striped`, `hover`, `condensed`, and `responsive` bootstrap options for table formatting.
- The first column (Gene) is made bold with a fixed width of 3 cm.
- The second column (Mutation Type) has a width of 4 cm.
- The third column (NumSamples) is colored blue.
- A footnote is added to summarize the content of the table.

Value

A styled table with mutation counts for selected genes, formatted for better presentation.

`visualize_cd8_markers` *Visualize CD8+ T cell Markers*

Description

This function visualizes CD8+ T cell markers using a FeaturePlot and identifies clusters enriched for these markers.

Usage

```
visualize_cd8_markers(
  seurat_obj,
  all_markers,
  cd8_markers,
  reduction_type = "umap",
  color_scheme = c("red", "blue")
)
```

Arguments

`seurat_obj`	A Seurat object.
`all_markers`	A data frame containing all identified markers.
`cd8_markers`	A vector of CD8+ T cell marker genes.
`reduction_type`	Type of dimensionality reduction to use for plotting (default is "umap").
`color_scheme`	A vector of colors for plotting (default is c("red", "blue")).

Value

Feature plot of CD8+ T cell markers and printed list of enriched clusters.

`visualize_cd8_subtypes` *Visualize CD8+ Subtypes on UMAP*

Description

This function visualizes the CD8+ T cell subtypes on a UMAP plot.

Usage

```
visualize_cd8_subtypes(seurat_obj)
```

Arguments

`seurat_obj`	A Seurat object containing CD8+ T cells with assigned labels.

Value

UMAP plot of CD8+ subtypes with custom theme.

`visualize_pca` *Visualize PCA and Elbow Plot*

Description

This function generates an Elbow plot for PCA variance and visualizes the PCA results using a DimPlot.

Usage

```
visualize_pca(seurat_obj)
```

Arguments

`seurat_obj`	A Seurat object containing single-cell RNA-seq data.

Value

The Elbow plot, PCA plot, and heatmap of the first 5 principal components.

visualize_umap *Visualize UMAP*

Description

This function visualizes the UMAP embedding of the Seurat object.

Usage

```
visualize_umap(seurat_obj)
```

Arguments

`seurat_obj`	A Seurat object containing UMAP results.

Value

UMAP plot with labels and custom themes.

Index