```
function subset()
    path = '/Users/sabrina/Library/CloudStorage/OneDrive-
UniversityofCopenhagen/Transcriptomics Data/Single Cell RNA 2';
    labelsFile = [path, '/filtered_neurons_doublets_labels.csv'];
    barcodesFile = [path, '/filtered_neurons_doublets_barcodes.tsv'];
    matrixFile = [path, '/filtered_neurons_doublets.csv'];
    classificationFile = '/Users/sabrina/Library/CloudStorage/OneDrive-
UniversityofCopenhagen/Thesis/CCC inference/NC/classification.csv';

    opts = detectImportOptions(labelsFile);
    opts = setvartype(opts, 'final_cluster_assignment', 'string');
    opts = setvartype(opts, 'cell_id', 'string');
    labels = readtable(labelsFile, opts);

    opts = detectImportOptions(barcodesFile, "FileType","text", 'Delimiter',
'\t');
    opts = setvartype(opts, 'barcode', 'string');
    barcodes = readtable(barcodesFile, opts);

    opts = detectImportOptions(classificationFile);
    opts = setvartype(opts, 'final_cluster_assignment', 'string');
    opts = setvartype(opts, 'cell_type', 'string');
    classification = readtable(classificationFile, opts);

    counts = readtable(matrixFile, "NumHeaderLines", 2);

    idx = ismember(labels.final_cluster_assignment,
classification.final_cluster_assignment);
    counts.Properties.VariableNames(1:3) = {'cell_idx', 'gene_idx', 'value'};

    labels_final = labels(idx, :);

    [unique_class_clusters, ia_class_unique] =
unique(classification.final_cluster_assignment, 'stable');
    cl_to_ct = containers.Map(unique_class_clusters,
classification.cell_type(ia_class_unique));

    cell_type_values = strings(height(labels_final), 1);
    for i = 1:height(labels_final)
        cluster_key = labels_final.final_cluster_assignment(i);
        if isKey(cl_to_ct, cluster_key)
            cell_type_values(i) = cl_to_ct(cluster_key);
        else
            cell_type_values(i) = "";
            disp('no corresponding cell type');
        end
    end
    labels_final.cell_type = cell_type_values;
    fprintf('%d cells before classification filtering.\n', height(labels));
    fprintf('%d cells kept after classification filtering.\n',
height(labels_final));
```

```
    barcodes_final = barcodes(idx, :);
    fprintf('%d barcodes kept.\n', height(barcodes_final));

    ind = find(idx);

    idx_counts = ismember(counts.cell_idx, ind);
    filtered_counts = counts(idx_counts, :);
    fprintf('%d non-zero entries kept.\n', height(filtered_counts));

    [lia, locb] = ismember(filtered_counts.cell_idx, ind);

    % lia should be all true here, because idx_counts already filtered for
members of ind.
    % locb will contain the 1-based indices: for each element X in
filtered_counts.cell_idx,
    % locb will give its position j such that ind(j) == X.

    if all(lia)
        filtered_counts.cell_idx = locb;
    end
    counts_final = filtered_counts;

    path = '/Users/sabrina/Library/CloudStorage/OneDrive-
UniversityofCopenhagen/Thesis/CCC inference/RNA_data_subset';
    if ~exist(path, 'dir')
        mkdir(path);
    end
    output_labels = fullfile(path, 'labels_subset.csv');
    output_barcodes = fullfile(path, 'barcodes.tsv');
    output_counts = fullfile(path, 'matrix.mtx');

    writetable(labels_final, output_labels);

    writetable(barcodes_final, output_barcodes, 'FileType', 'text',
'Delimiter', '\t', 'WriteVariableNames',false);

    fid = fopen(output_counts, 'w');
    fprintf(fid, '%%%%MatrixMarket matrix coordinate integer general\n');
    fprintf(fid, '%d    %d  %d\n',
counts_final{height(counts_final),"gene_idx"}, height(labels_final),
height(counts_final));
    fclose(fid);

    if all(ismember({'gene_idx', 'cell_idx', 'value'},
counts_final.Properties.VariableNames))
        data_to_write = [counts_final.gene_idx, counts_final.cell_idx,
counts_final.value];
        dlmwrite(output_counts, data_to_write, '-append', 'delimiter', '\t',
'precision', '%d');
    end
end
```

```
28238 cells before classification filtering.
3126 cells kept after classification filtering.
```

```
3126 barcodes kept.
7382333 non-zero entries kept.
```

```
clear
subset()
```

Published with MATLAB® R2024b