Homework 4

Priyanka Emani - 1001981861
Aravind Kashyap - 1001956591
Shreyas Jagadeep - 1001888859

UNIVERSITY OF TEXAS ARLINGTON

CSE: 5370 Bioinformatics

April 27, 2022

Collaboration Statement

The assignment was carried out as a group. Following students were part of the group and made individual contributions in completing the assignment.

1. Bioinformatics

Single Cell RNA Analysis. The dataset used describes single-cell mouse tissues. The dataset contains 100,000 cells from the brain of a mouse. Each column in the expression matrix csv file corresponds to a a transcript (gene) wherelese each row corresponds to a single cell. The metadata csv file describes each cell.

2. Imports and Data Objects

```python
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import scanpy as sc
import pandas as pd
import numpy as np
%matplotlib inline

count = pd.read_csv('/content/drive/MyDrive/data/brain_counts.csv', index_col=0)
metadata = pd.read_csv('/content/drive/MyDrive/data/brain_metadata.→
,csv',index_col=0)
adata = sc.AnnData(X = count, obs = metadata)
```

3. Preprocessing

1.0 Spiked genes

```python
# record and keep count of spikes
is_spiked = {}
num_spikes = 0
for gene in adata.var_names:
if 'ERCC' in gene:
is_spiked[gene] = True
num_spikes += 1
else:
is_spiked[gene] = False
adata.var['ERCC'] = pd.Series(is_spiked)
# write
adata.write('/content/drive/MyDrive/data/brain_raw.h5ad')
# read
adata = sc.read('/content/drive/MyDrive/data/brain_raw.h5ad')
```

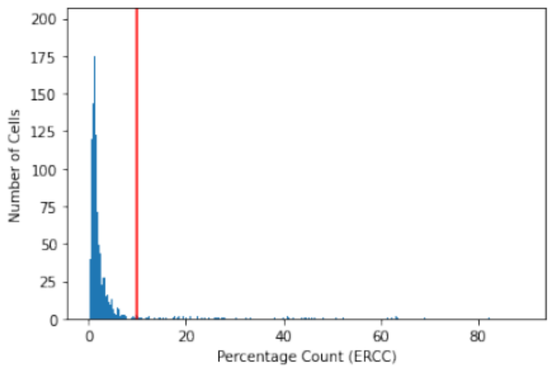
2.0 Quality control

```python
# Compute QC Metrics
qc = sc.pp.calculate_qc_metrics(adata, qc_vars = ['ERCC'])
cell_qc = qc[0]
gene_qc = qc[1]
```

qc - cells

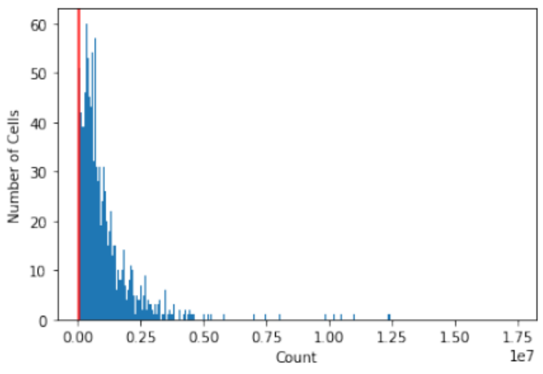
```python


# visualize spike-ins
# outliers to be filtered
plt.hist(cell_qc['pct_counts_ERCC'], bins=1000)
plt.xlabel('Percentage Count (ERCC)')
plt.ylabel('Number of Cells')
plt.axvline(10, color='red')
```

<matplotlib.lines.Line2D at 0x7fdffb780c50>



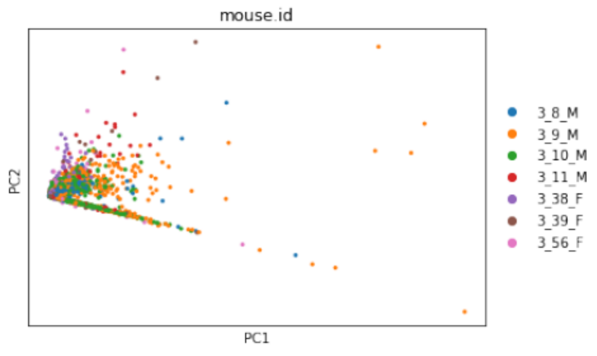
```python
# visualize lib size (reads per cell)
# cell with few reads to be filtered
plt.hist(cell_qc['total_counts'], bins=1000)
plt.xlabel('Count')
plt.ylabel('Number of Cells')
plt.axvline(50000, color='red')
```

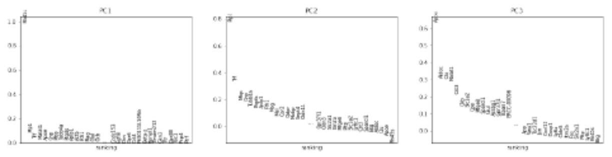
<matplotlib.lines.Line2D at 0x7fdffad67710>

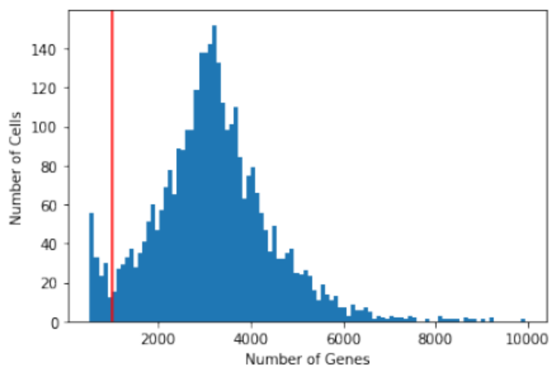


```python
# visualize detected gene-count (in cells)
# outliers to be filtered
plt.hist(cell_qc['n_genes_by_counts'], bins=100)
plt.xlabel('Number of Genes')
plt.ylabel('Number of Cells')
plt.axvline(1000, color='red')
```

<matplotlib.lines.Line2D at 0x7fdffa2f8150>



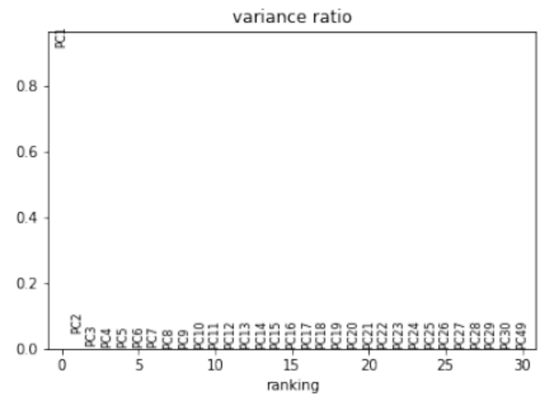


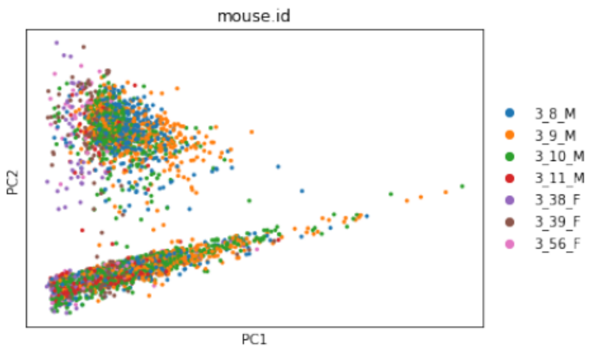


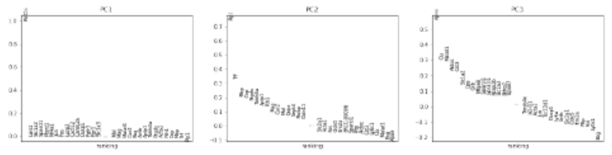
```
# filtering
# by ERCC
low_ERCC = (cell_qc['pct_counts_ERCC'] < 10)
adata = adata[low_ERCC]
# by Gene count
sc.pp.filter_cells(adata, min_genes = 1000)
# write
adata.write('/content/drive/MyDrive/data/brain_qc.h5ad')
# read
adata = sc.read('/content/drive/MyDrive/data/brain_qc.h5ad')
```

3.0 Normalization
Cell library normalization

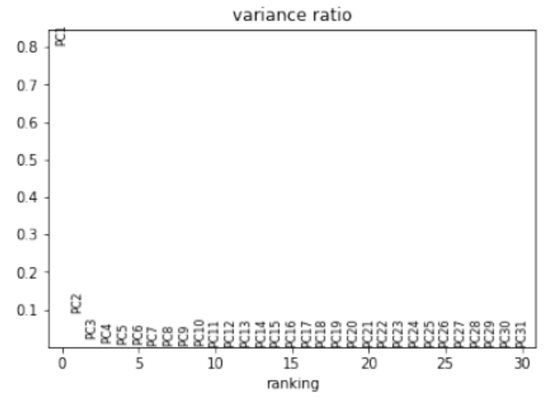
```

# visualization of data before Normalization
sc.pp.pca(adata)
sc.pl.pca_overview(adata, color='mouse.id')
```

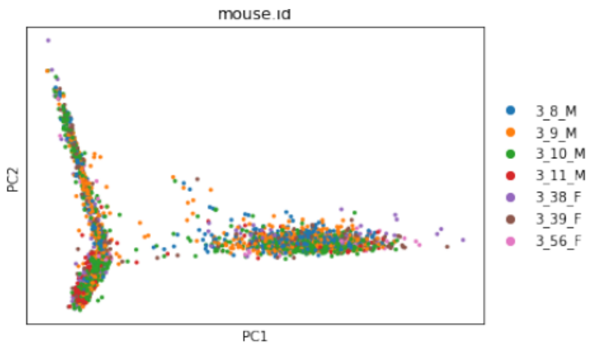


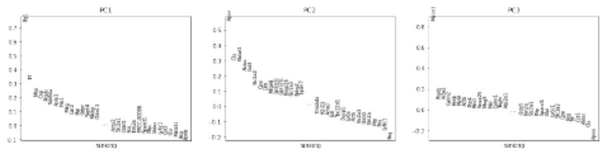




```python
  # copy data. copy and original to be used for comparison
adata_ = adata.copy()
# copy data before normalizing
adata_.raw = adata_
# CPM normalization
sc.pp.normalize_per_cell(adata_,counts_per_cell_after=1e6)
# visualization of data after Normalization
sc.pp.pca(adata_)
sc.pl.pca_overview(adata_, color='mouse.id')

```





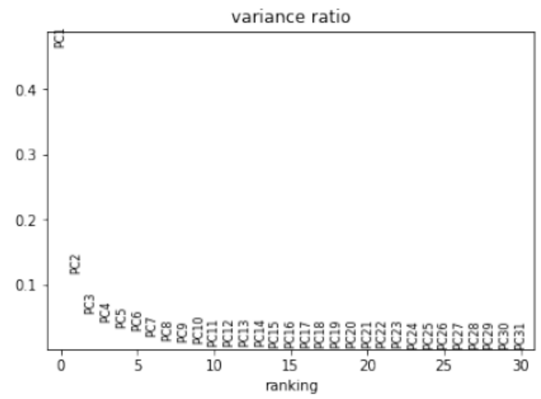


Gene expression normalization

```
# Rn45s gene is dorminant in data.
# Use normalization to deal with imbalance
Rn45s_0 = adata_.var.index != 'Rn45s'
adata_Rn45s_0 = adata_[:, Rn45s_0]
# visualization of data after Normalization
sc.pp.pca(adata_Rn45s_0)
sc.pl.pca_overview(adata_Rn45s_0, color='mouse.id')

```

```
        # centering and scaling gene expression data
sc.pp.log1p(adata_)
sc.pp.scale(adata_)
# visualization after Normalization
sc.pp.pca(adata_)
sc.pl.pca_overview(adata_, color='plate.barcode')
```

```
# write
```


variance ratio
ranking

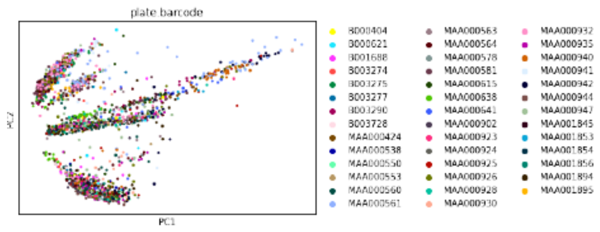
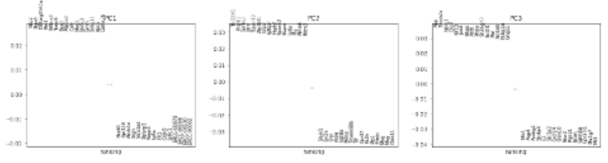
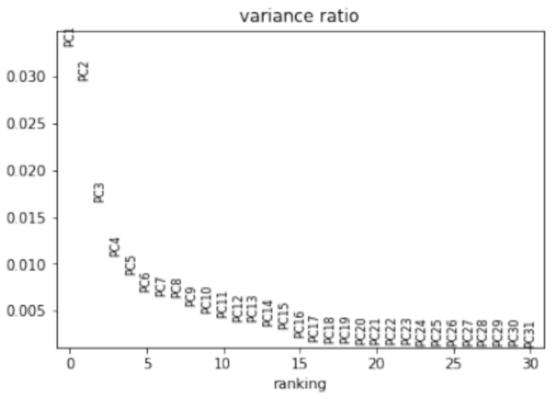
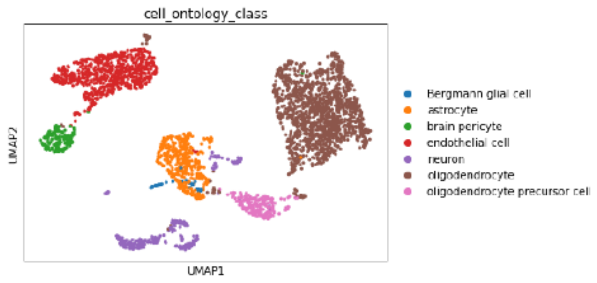
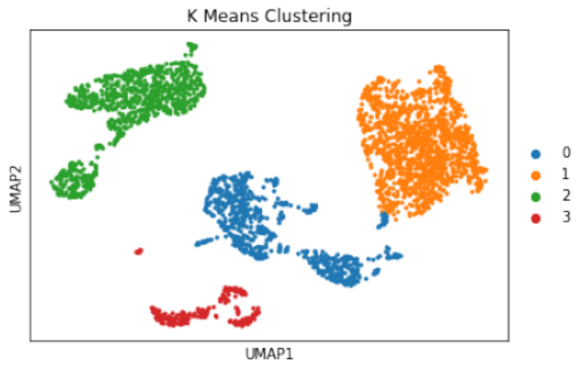


plate barcode
PC1
PC2
B000404
B000621
B001688
B003274
B003275
B003277
B003290
B003728
MAA000424
MAA000538
MAA000550
MAA000553
MAA000560
MAA000561
MAA000563
MAA000564
MAA000578
MAA000581
MAA000615
MAA000638
MAA000641
MAA000902
MAA000923
MAA000924
MAA000925
MAA000926
MAA000928
MAA000930
MAA000932
MAA000935
MAA000940
MAA000941
MAA000942
MAA000944
MAA000947
MAA001845
MAA001853
MAA001854
MAA001856
MAA001894
MAA001895


PC1
PC2
PC3
ranking


variance ratio
ranking





```
adata_.write('/content/drive/MyDrive/data/brain_normalized.h5ad')
# read
adata = sc.read('/content/drive/MyDrive/data/brain_normalized.h5ad')
```

4. Analysis

Dimensionality Reduction -UMAP

```
sc.pp.neighbors(adata)
sc.tl.umap(adata, min_dist=0.5, spread=1.0, n_components=2)
# UMAP visualization using Dimensionality Reduction
sc.pl.umap(adata, color='cell_ontology_class')
```

Clustering - UMAP

```

umap_= adata.obsm['X_umap']
kmeans = KMeans(n_clusters=4).fit(umap_)
adata.obs['K Means Clustering'] = kmeans.labels_
adata.obs['K Means Clustering'] = adata.obs['K Means Clustering'].astype(str)
# UMAP visualization using KMeans Clsutering
sc.pl.umap(adata, color='K Means Clustering')
```

References

Below are the list of references we used for this project.
1. https://github.com/scverse/scanpy-tutorials/issues/28
2. https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSM3109048
3. https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE49712
4. https://scanpy.readthedocs.io/en/stable/generated/scanpy.pp.calculate$_q c_m etrics.html$
5.$https : //github.com/scverse/scanpy/issues/978$
6.$https : //github.com/scverse/scanpy/issues/1147$
7.$https : //www.oreilly.com/library/view/python - data - science/9781491912126/ch04.html$
8.$https : //matplotlib.org/stable/tutorials/introductory/pyplot.html$
9.$https : //chanzuckerberg.github.io/scRNA-python-workshop/preprocessing/01-basic-qc.html$
10.$https : //github.com/AmoDinho/datacamp-python-data-science-track/blob/master/Machine$
11.$https : //scanpy.readthedocs.io/en/stable/generated/scanpy.pp.filter_c ells.html$
12.$https : //scanpy.readthedocs.io/en/stable/generated/scanpy.pl.pca.html$
13.$https : //github.com/scverse/scanpy/issues/324$
14.$https : //nbisweden.github.io/workshop-scRNAseq/labs/compiled/scanpy/scanpy_0 4_c lustering.html$
15.$https : //scanpy.readthedocs.io/en/stable/generated/scanpy.pp.normalize_p er_c ell.html$
16.$https : //scanpy.readthedocs.io/en/stable/generated/scanpy.pp.normalize_t otal.html$
17.$https : //www.ncbi.nlm.nih.gov/pmc/articles/PMC4404308/$
19.$https : //scanpy.readthedocs.io/en/stable/generated/scanpy.tl.umap.html$
20.$https : //scanpy.readthedocs.io/en/stable/generated/scanpy.pp.neighbors.html$
21.$https : //github.com/scverse/scanpy/blob/master/scanpy/tools/_u map.py$
22.$https : //github.com/theislab/scvelo/issues/37$
23.$https : //towardsdatascience.com/umap - and - k - means - to - classify - characters - league - of - legends - 668a788cb3c1$
24.$https : //umap - learn.readthedocs.io/en/latest/clustering.html$
25.$https : //scanpy.readthedocs.io/en/stable/generated/scanpy.pp.log1p.html$

Also we have taken help from Dr. Steven Fernandes who is post doctoral research at University of Central Florida