Dataset GSM3215435

Loading data

Ввод [2]:

```
import numpy as np
import pandas as pd
import scanpy as sc
from scipy.io import mmread
import anndata as ad
import pandas as pd

barcodes = pd.read_csv('GSM3215435/GSM3215435_ldlr_ko_barcodes.tsv.gz',
                       header = None, names = ['cells'])
genes = pd.read_csv('GSM3215435/GSM3215435_ldlr_ko_features.tsv.gz', delimiter = '\t',
                       header = None, names = ['gene_name', 'gene_symbol'])
matrix = mmread('GSM3215435/GSM3215435_ldlr_ko_matrix.mtx.gz').tocsr()
```

Ввод [2]:

```
adata = ad.AnnData(matrix.T, var=genes, obs=barcodes)
```

```
C:\Users\yulia\AppData\Local\Temp\ipykernel_20496\4018781955.py:1: FutureWar
ning: X.dtype being converted to np.float32 from int64. In the next version
of anndata (0.9) conversion will not be automatic. Pass dtype explicitly to
avoid this warning. Pass `AnnData(X, dtype=X.dtype, ...)` to get the future
behavour.
  adata = ad.AnnData(matrix.T, var=genes, obs=barcodes)
C:\Users\yulia\AppData\Local\Programs\Python\Python38\scanpy\lib\site-packag
es\anndata\_core\anndata.py:121: ImplicitModificationWarning: Transforming t
o str index.
  warnings.warn("Transforming to str index.", ImplicitModificationWarning)
```

Ввод [3]:

```
sc.pp.filter_cells(adata, min_genes=20)
sc.pp.filter_genes(adata, min_cells=20)
```

Ввод [4]:

```
adata
```

Out[4]:

```
AnnData object with n_obs × n_vars = 3781 × 12145
    obs: 'cells', 'n_genes'
    var: 'gene_name', 'gene_symbol', 'n_cells'
```

Ввод [5]:

```
adata.var_names = adata.var['gene_symbol']
adata.var
```

Out[5]:

gene_symbol	gene_name	gene_symbol	n_cells
Mrpl15	ENSMUSG00000033845	Mrpl15	1349
Lypla1	ENSMUSG00000025903	Lypla1	1181
Tcea1	ENSMUSG00000033813	Tcea1	1863
Atp6v1h	ENSMUSG00000033793	Atp6v1h	1329
Rb1cc1	ENSMUSG00000025907	Rb1cc1	917
...	...	...	...
AC125149.2	ENSMUSG00000079794	AC125149.2	26
AC168977.1	ENSMUSG00000079808	AC168977.1	65
PISD	ENSMUSG00000095041	PISD	2263
DHRSX	ENSMUSG00000063897	DHRSX	1017
CAAA01147332.1	ENSMUSG00000095742	CAAA01147332.1	39

12145 rows × 3 columns

Ввод [6]:

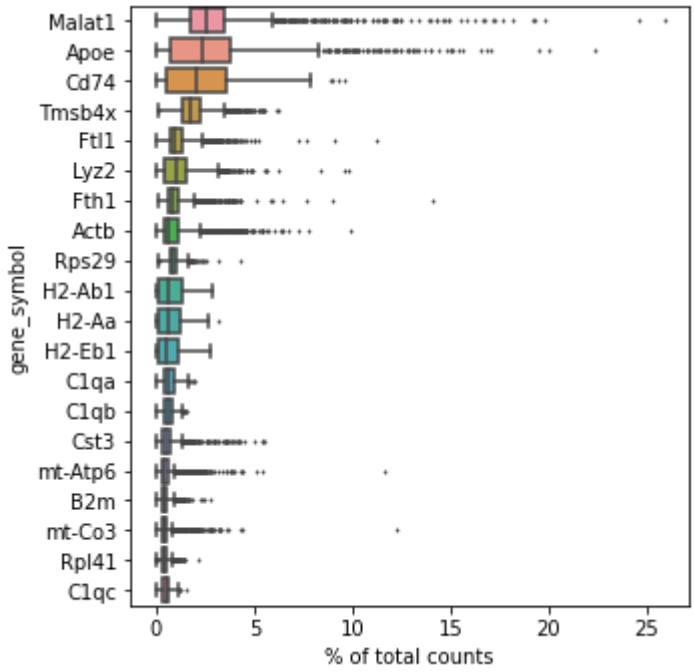
```
adata.obs.head(10)
```

Out[6]:

	cells	n_genes
0	AAACCTGAGATGCCTT-1	2218
1	AAACCTGAGCTAGTGG-1	1209
2	AAACCTGCAAGCTGTT-1	2084
3	AAACCTGCACCGAATT-1	1961
4	AAACCTGCACGCTTTC-1	908
5	AAACCTGCAGGACCCT-1	2985
6	AAACCTGGTCACAAGG-1	2028
7	AAACCTGGTCACCTAA-1	1727
8	AAACCTGGTCCGTGAC-1	3000
9	AAACCTGGTCTCACCT-1	1156

Ввод [7]:

```
sc.pl.highest_expr_genes(adata, n_top=20, )
```



Visualization and filtering out poor-quality cells

Ввод [8]:

```
adata.var['mt'] = adata.var_names.str.startswith('mt-')
adata.var['mt'].value_counts()
```

Out[8]:

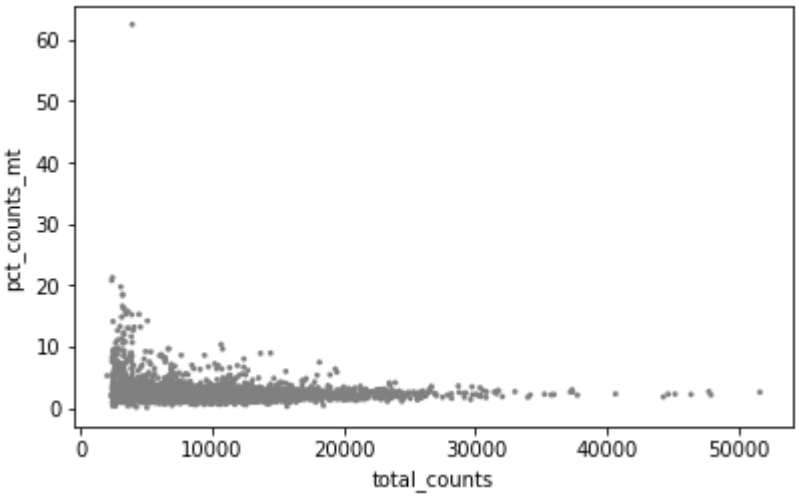
```
False    12132
True        13
Name: mt, dtype: int64
```

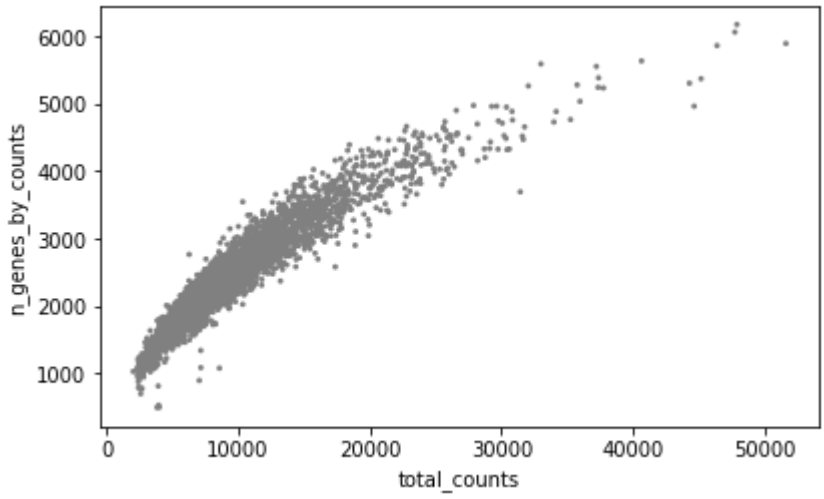
Ввод [9]:

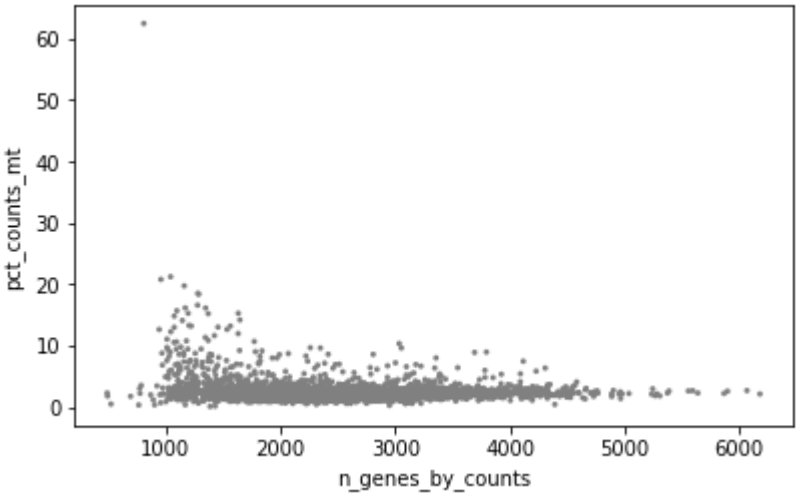
```
sc.pp.calculate_qc_metrics(adata, qc_vars=['mt'], percent_top=None, log1p=True, inplace=Tru
```

Ввод [10]:

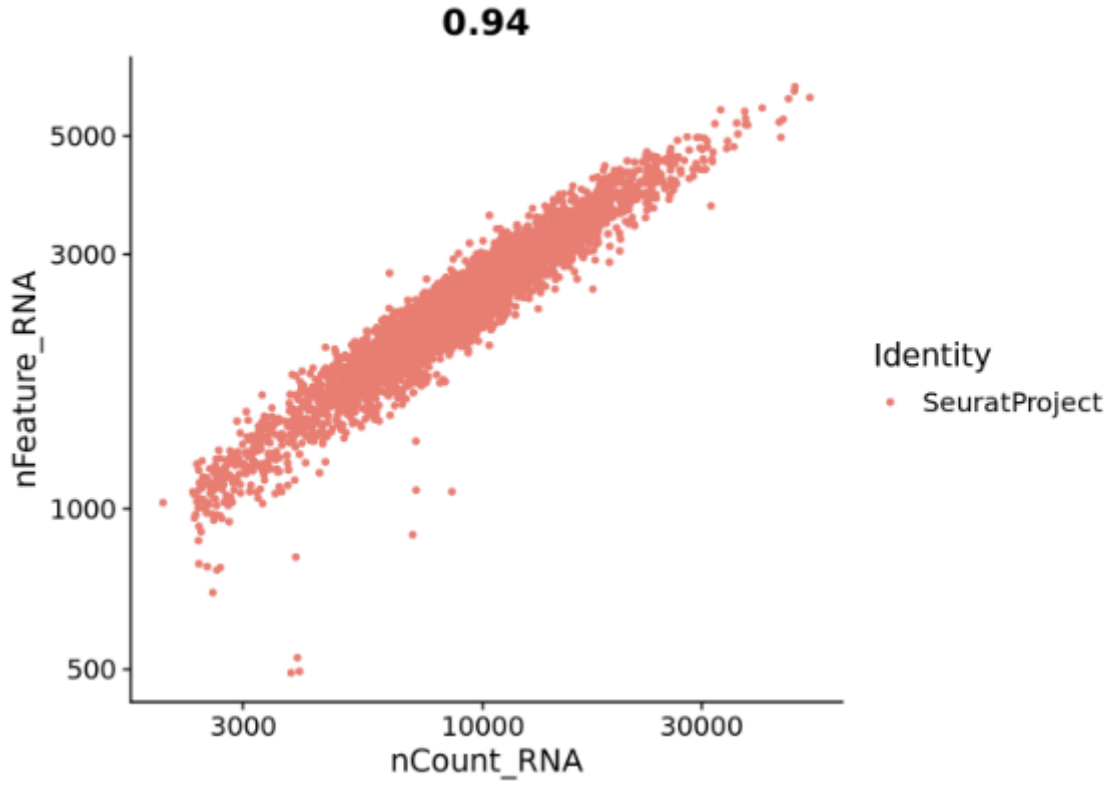
```
sc.pl.scatter(adata, x='total_counts', y='pct_counts_mt')
sc.pl.scatter(adata, x='total_counts', y='n_genes_by_counts')
sc.pl.scatter(adata, x='n_genes_by_counts', y='pct_counts_mt')
```

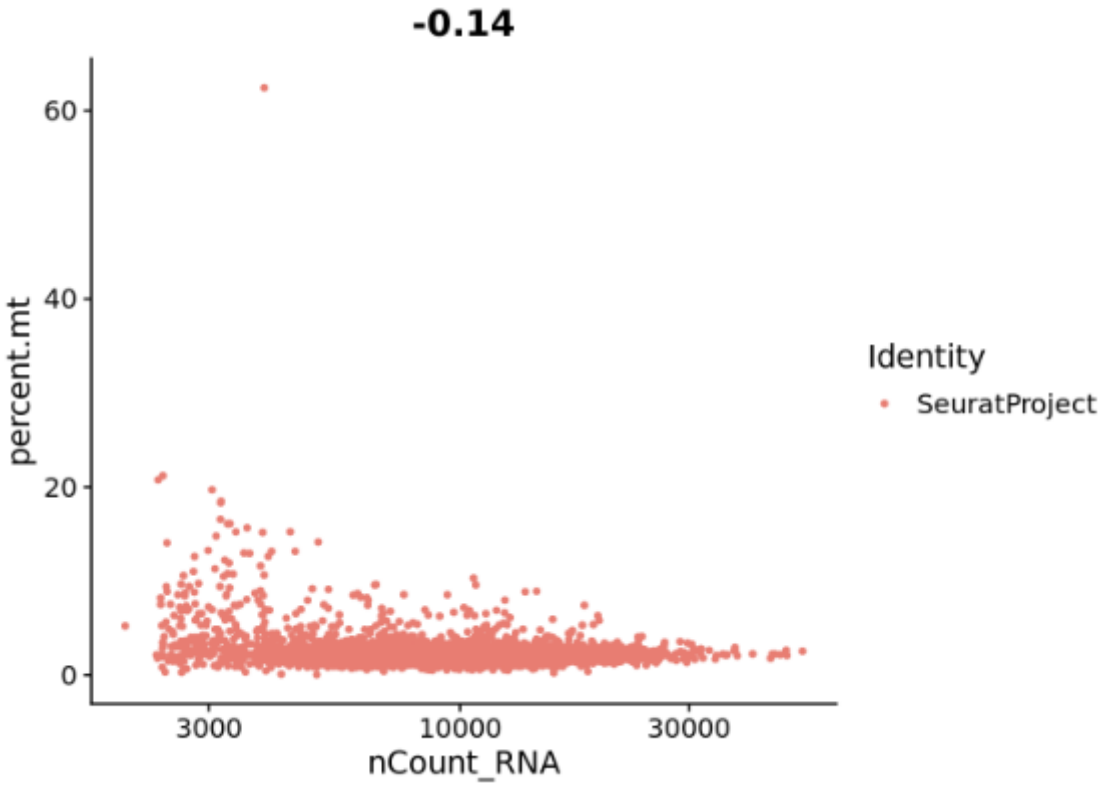


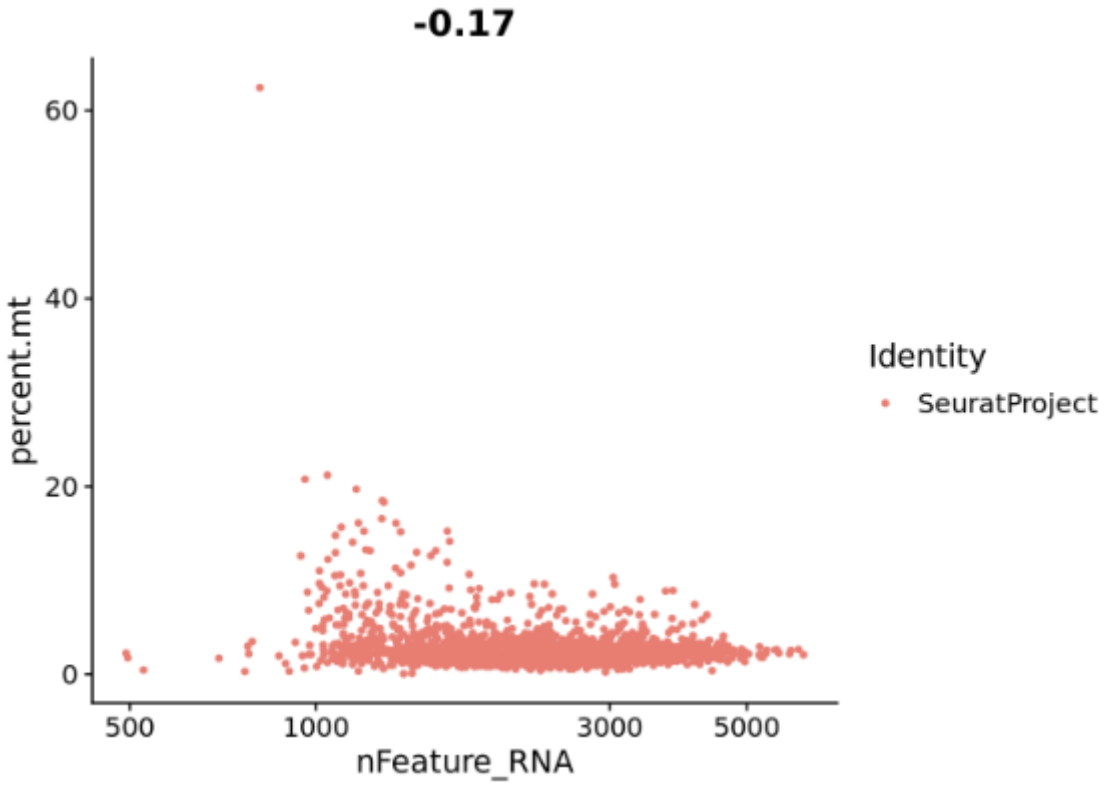




Plots are similar to my plots from seurat:

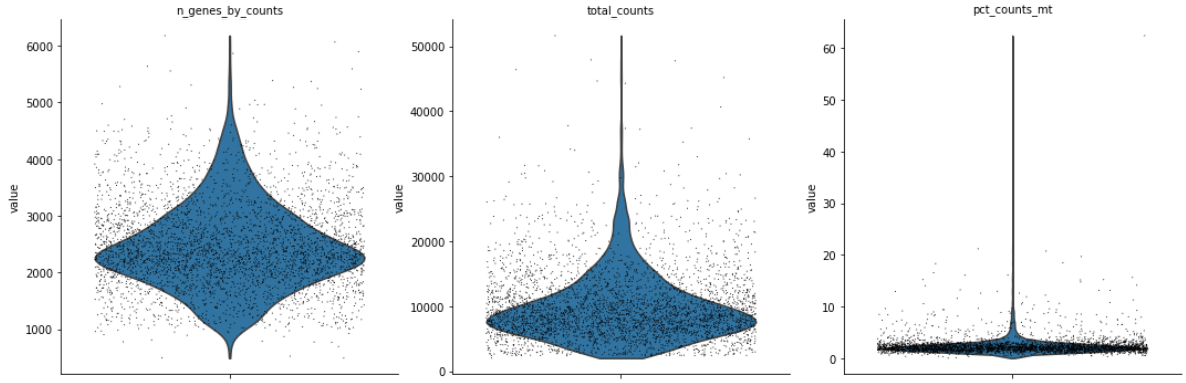



-0.14
percent.mt
60
40
20
0
3000
10000
30000
nCount_RNA
Identity
SeuratProject


-0.17
percent.mt
60
40
20
0
500
1000
3000
5000
nFeature_RNA
Identity
SeuratProject

Ввод [11]:

```
adata.var_names_make_unique()
sc.pl.violin(adata, ['n_genes_by_counts', 'total_counts', 'pct_counts_mt'],
             jitter=0.4, multi_panel=True)
```



Ввод [12]:

```
adata = adata[adata.obs.n_genes_by_counts > 1000, :].copy()
adata = adata[adata.obs.pct_counts_mt < 5, :].copy()
```

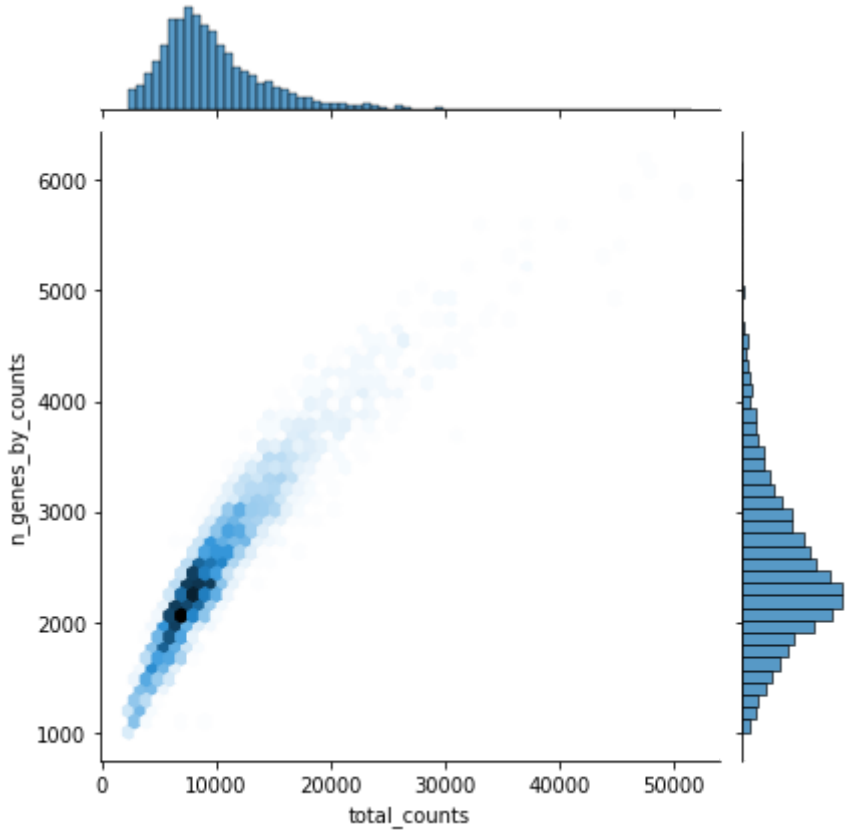
Ввод [13]:

```
import scanpy as sc
import seaborn as sns

sns.jointplot(
    data=adata.obs,
    x="total_counts",
    y="n_genes_by_counts",
    kind="hex",
)
```

Out[13]:

```
<seaborn.axisgrid.JointGrid at 0x122e0cebc70>
```



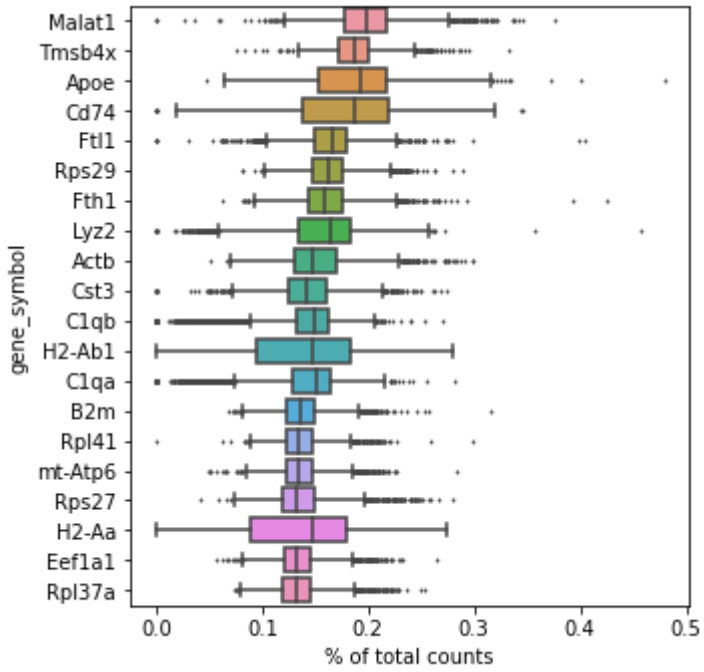
Data normalization and scaling

Ввод [14]:

```
sc.pp.normalize_total(adata, target_sum=1e4)
sc.pp.log1p(adata)
```

Ввод [15]:

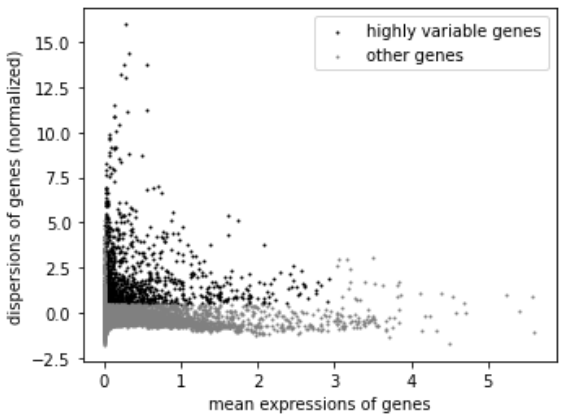
```
sc.pl.highest_expr_genes(adata, n_top=20, )
```

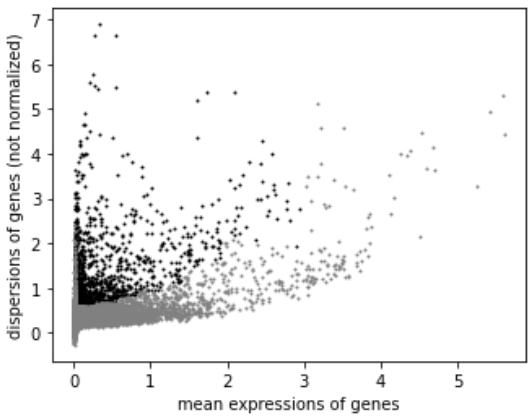


Identification of highly variable features

Ввод [16]:

```
import matplotlib.pyplot as plt
sc.pp.highly_variable_genes(adata, min_mean=0.0125, max_mean=3, min_disp=0.5)
gca = sc.pl.highly_variable_genes(adata, show=False, log=False)
```





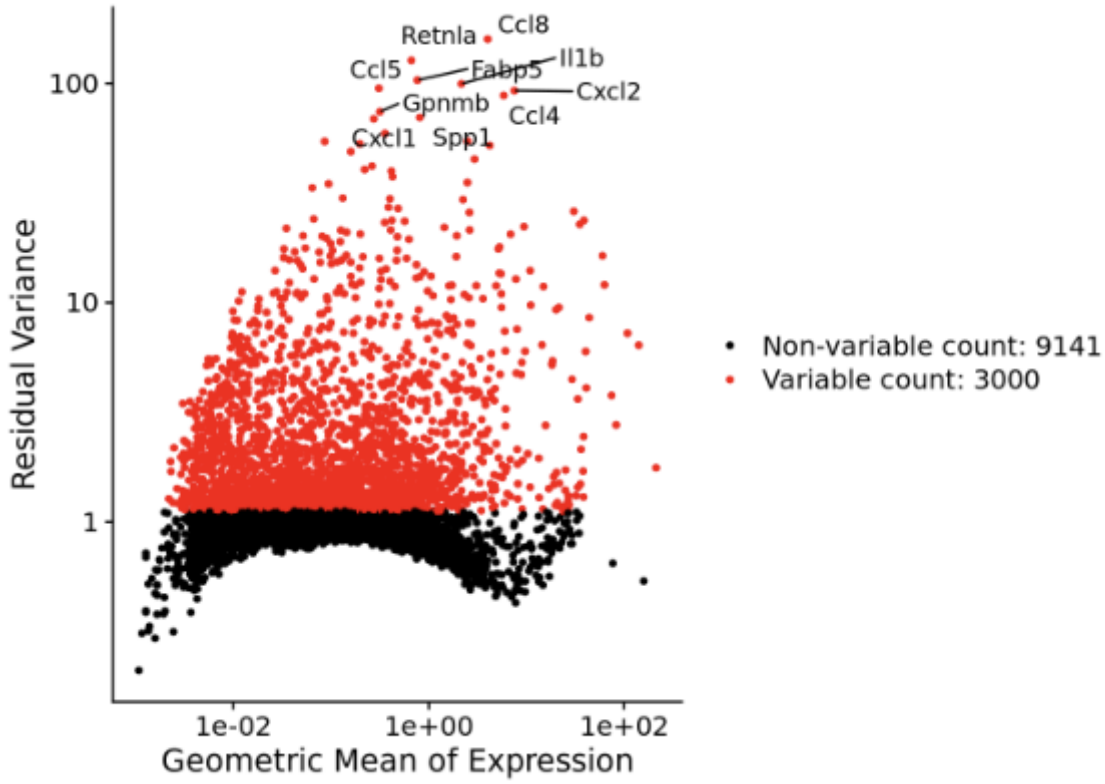
Ввод [17]:

```
adata.var['highly_variable'].value_counts()
```

Out[17]:

```
False    10904
True      1241
Name: highly_variable, dtype: int64
```

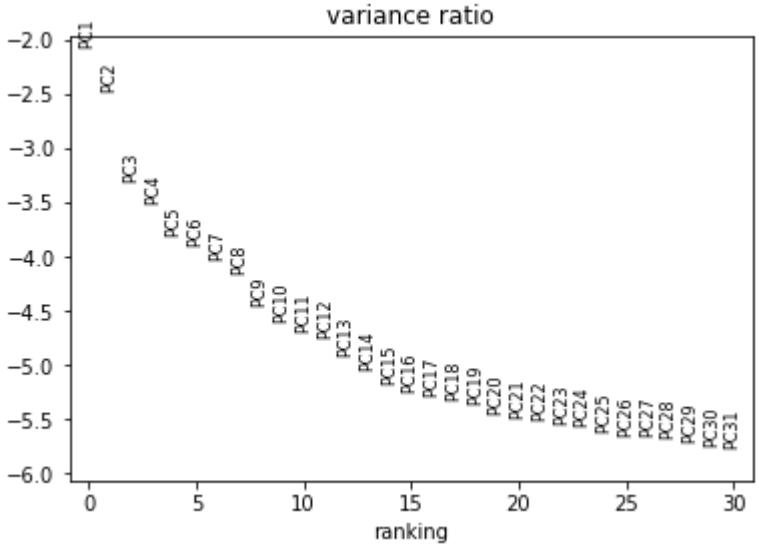
Volcano plot from seurat



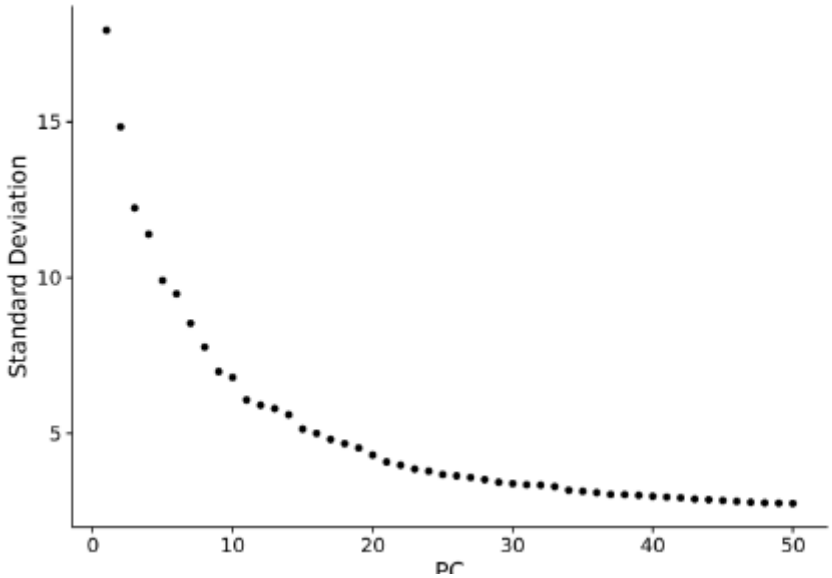
PCA

Ввод [18]:

```
sc.tl.pca(adata, svd_solver='arpack')
sc.pl.pca_variance_ratio(adata, log=True)
```



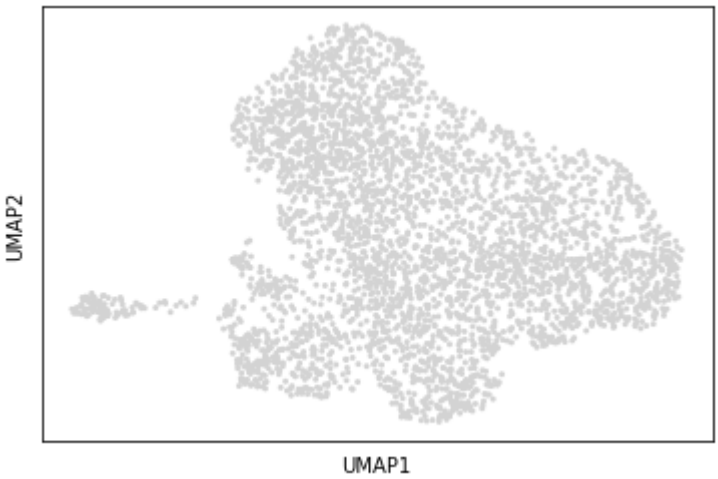
PCA plot from seurat



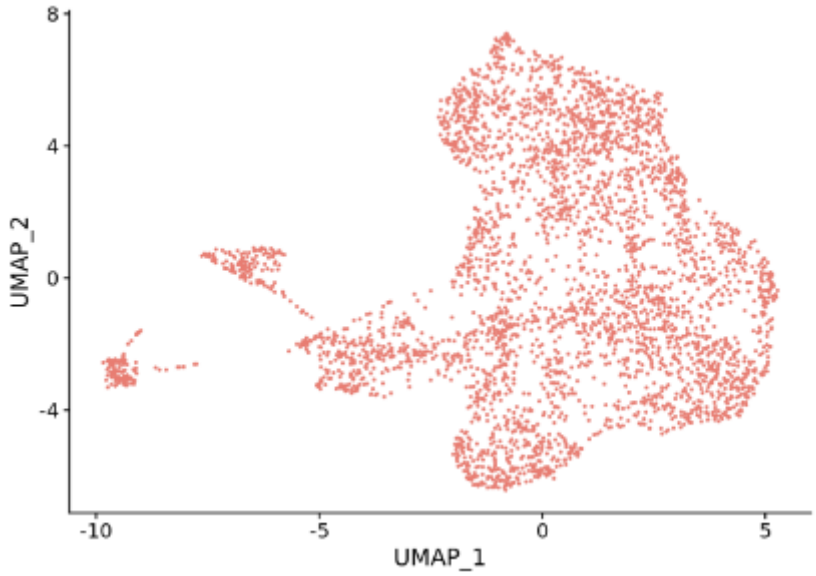
UMAP

Ввод [19]:

```
sc.pp.neighbors(adata, n_neighbors=30, n_pcs=10)
sc.tl.umap(adata)
sc.pl.umap(adata)
```

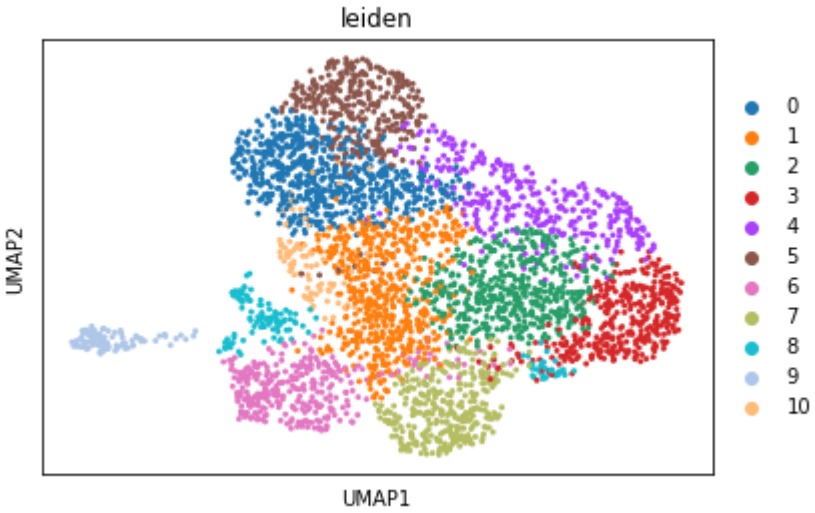


UMAP plot from seurat

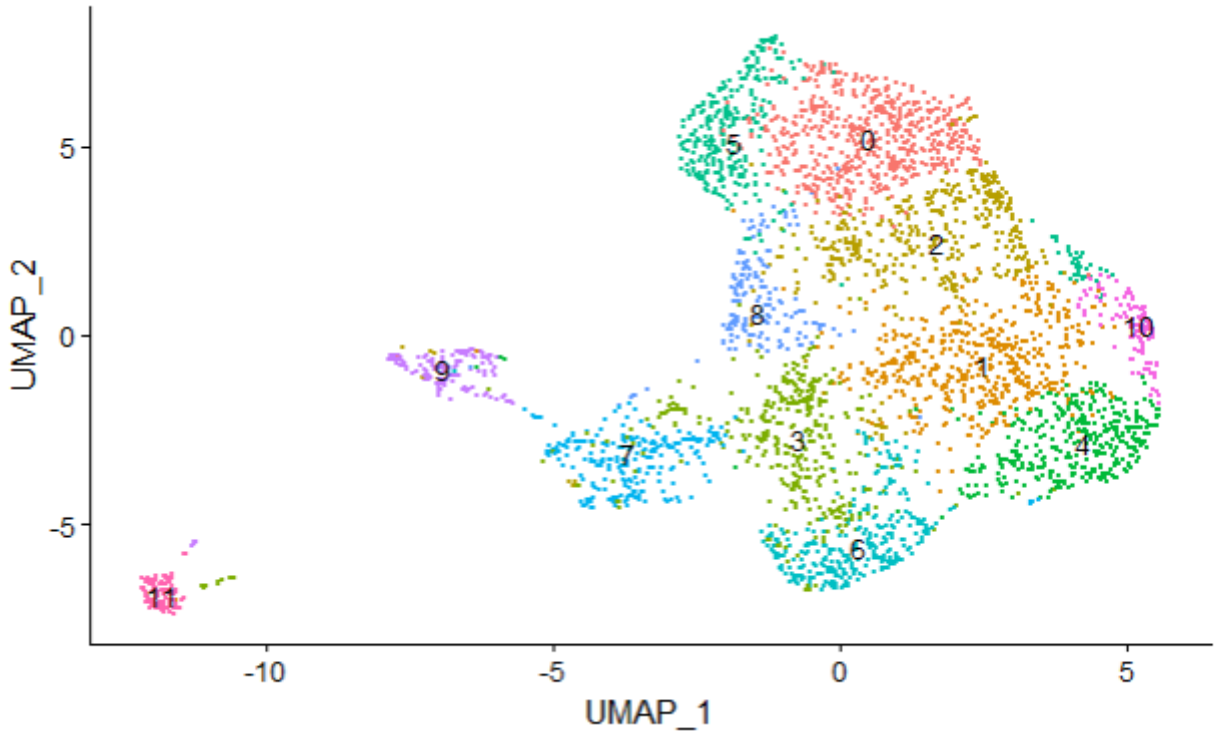


Ввод [20]:

```
sc.tl.leiden(adata, resolution=0.6)
sc.pl.umap(adata, color='leiden')
```



Clusters from seurat



T-SNE (authors use T-SNE method in publication)

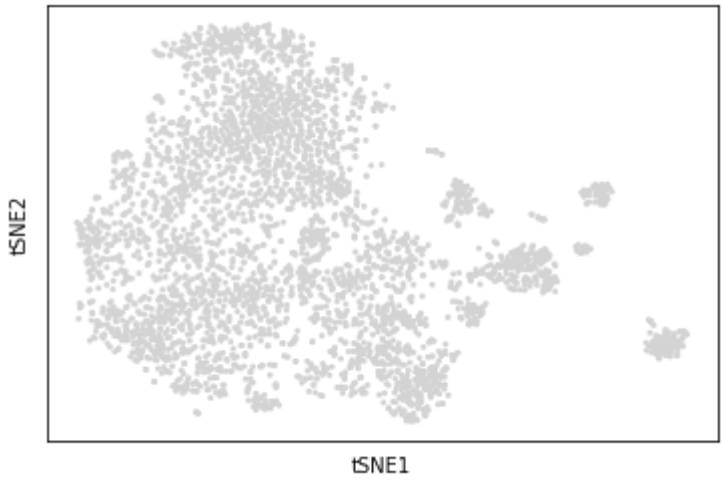
Ввод [21]:

```
sc.pp.neighbors(adata, n_neighbors=10, n_pcs=10)
sc.tl.tsne(adata)
```

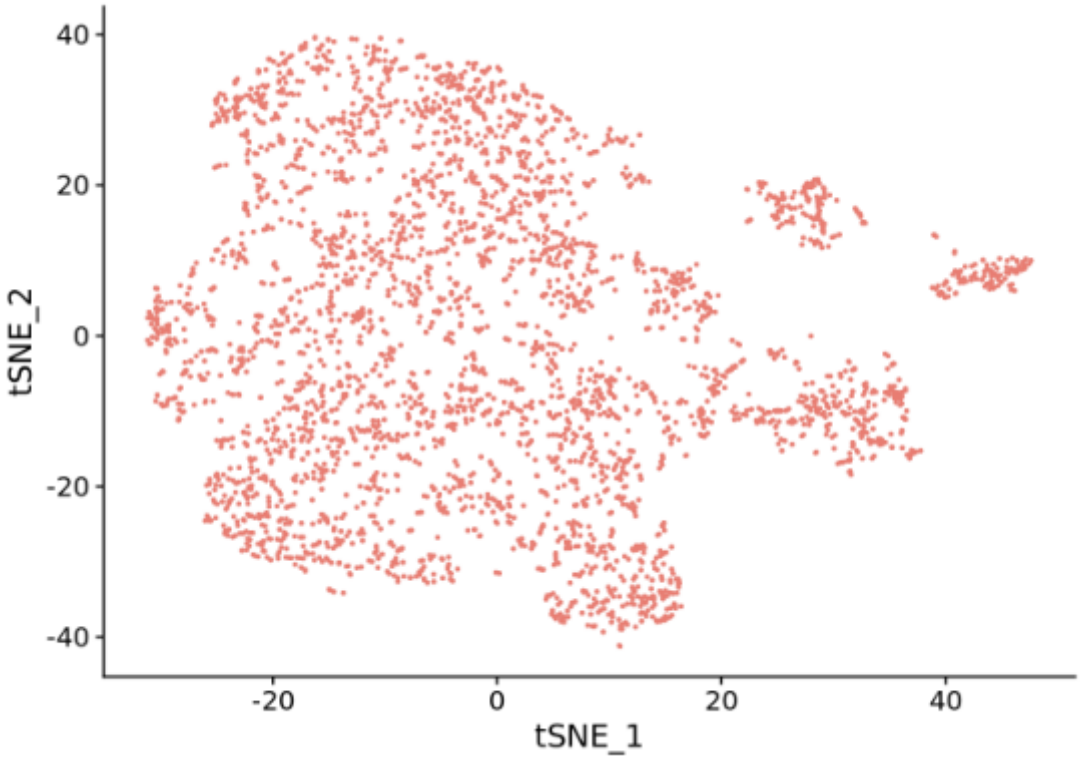
```
C:\Users\yulia\AppData\Local\Programs\Python\Python38\scanpy\lib\site-packag
es\sklearn\manifold\_t_sne.py:795: FutureWarning: The default initialization
in TSNE will change from 'random' to 'pca' in 1.2.
  warnings.warn(
```

Ввод [22]:

```
sc.pl.tsne(adata)
```

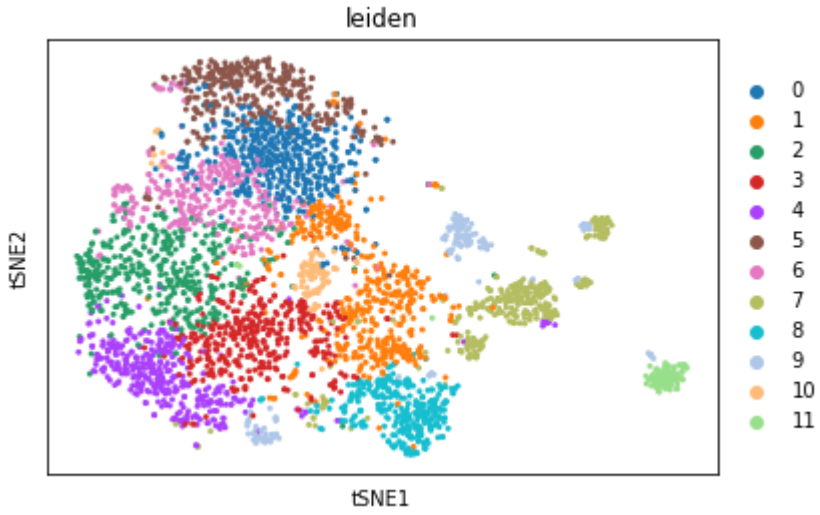


T-SNE plot from Seurat

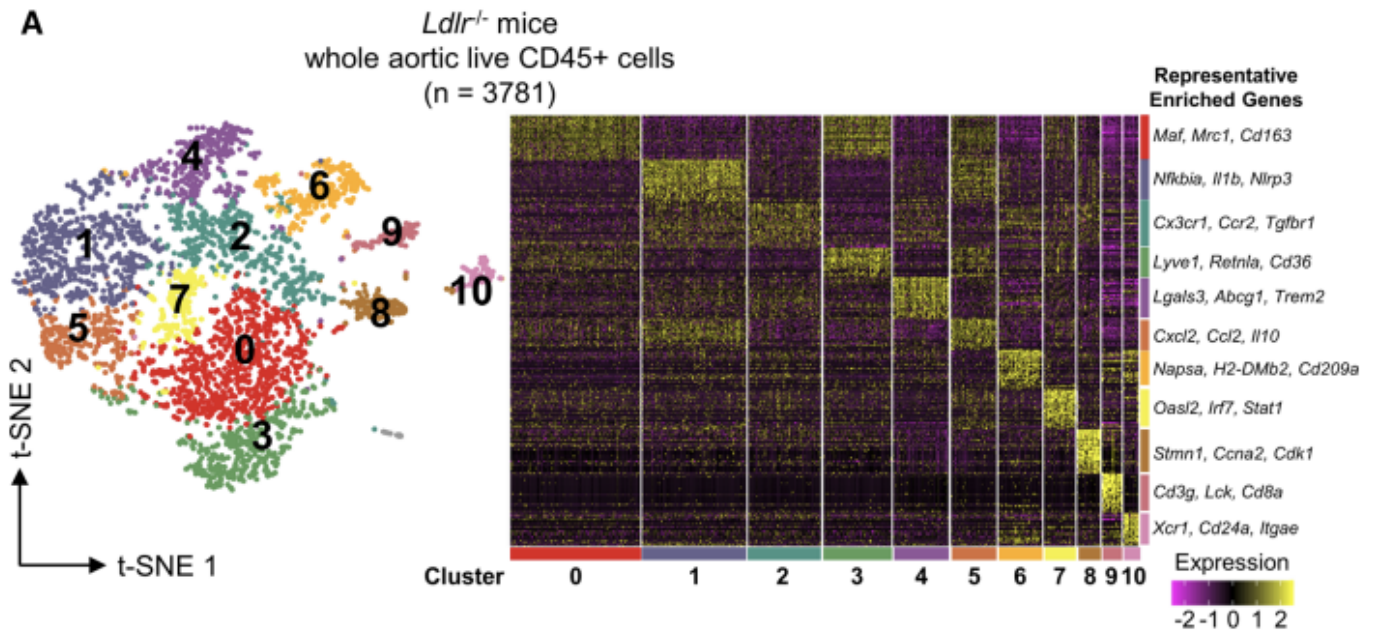


Ввод [23]:

```
sc.tl.leiden(adata, resolution=0.6)
sc.pl.tsne(adata, color='leiden')
```



T-SNE plot and markers from publication



Detection of cluster markers

Ввод [24]:

```
sc.tl.rank_genes_groups(adata, 'leiden', method='wilcoxon')
pd.DataFrame(adata.uns['rank_genes_groups']['names']).head(1)
```

Out[24]:

	0	1	2	3	4	5	6	7	8	9	10	11
0	Pf4	Cx3cr1	Junb	Atf3	Ifrd1	Folr2	Jun	Napsa	Lgals3	Stmn1	Bst2	Rps15a

Markers detceted by Seurat "Pf4" "Ccl4" "Hspa1a" "Plac8" "Il1b" "Retnla" "Spp1" "Ifitm1" "Isg15" "Stmn1" "Ccl7" "Ccl5"

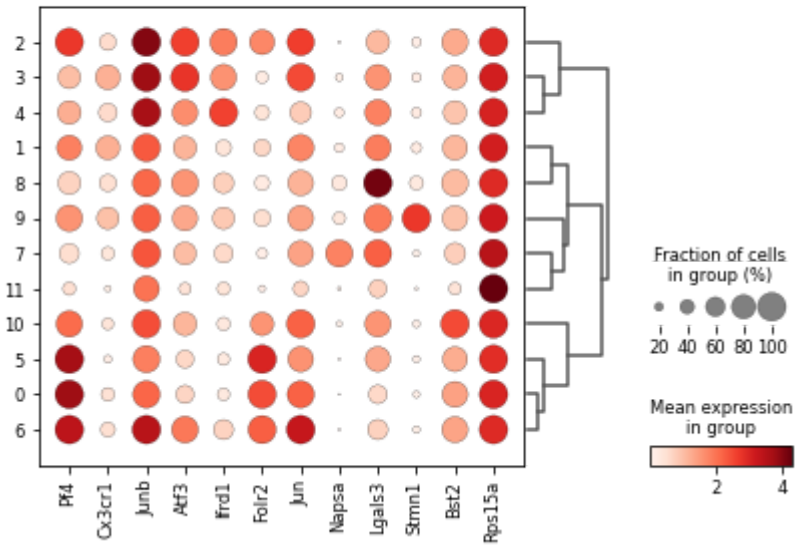
Common detected markers "Pf4" "Il1b" "Stmn1"

Expression of marker genes among clusters

Ввод [25]:

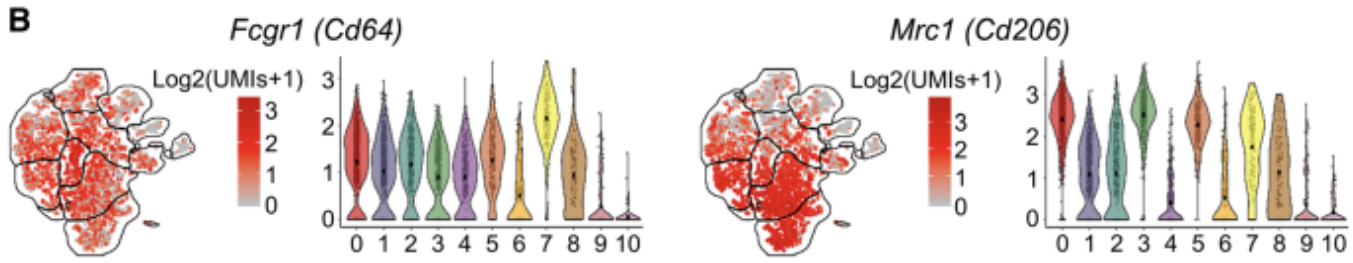
```
markers = ['Pf4', 'Cx3cr1','Junb', 'Atf3', 'Ifrd1','Folr2', 'Jun', 'Napsa', 'Lgals3', 'Stmn
sc.pl.dotplot(adata, markers, 'leiden', dendrogram=True)
```

```
WARNING: dendrogram data not found (using key=dendrogram_leiden). Running `s
c.tl.dendrogram` with default parameters. For fine tuning it is recommended
to run `sc.tl.dendrogram` independently.
```



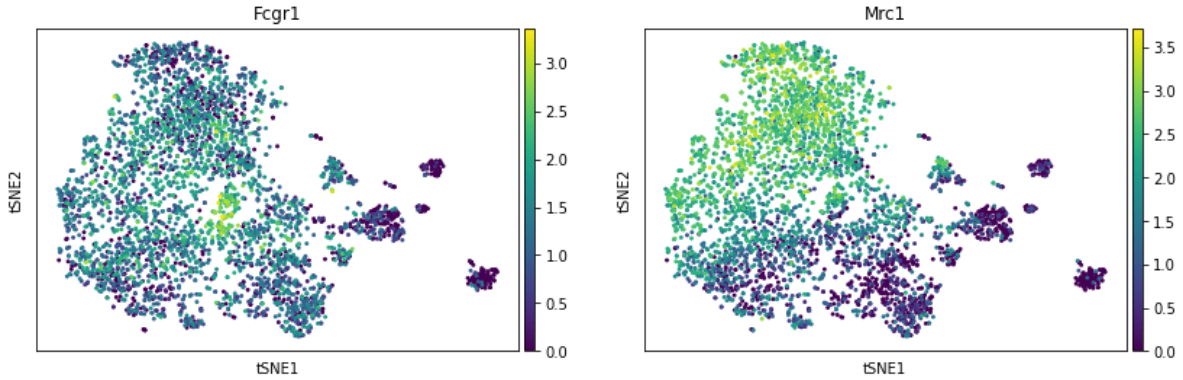
Expression of principal hematopoietic markers in the identified cell clusters (comparison between the publication and my results)

"Fcgr1", "Mrc1" expression



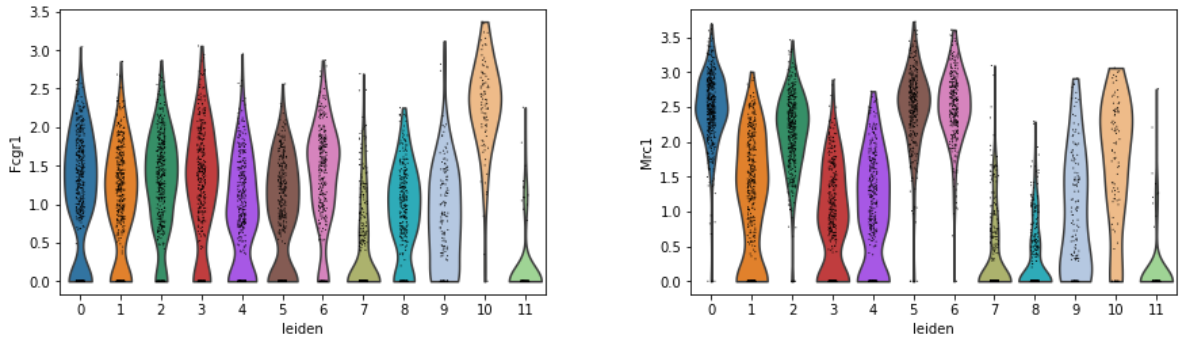
Ввод [26]:

```
sc.pl.tsne(adata, color=["Fcgr1", "Mrc1"])
```

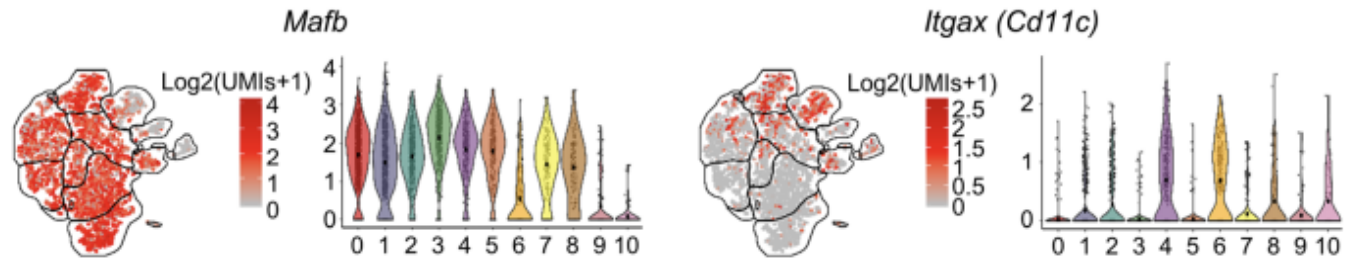


Ввод [27]:

```
sc.pl.violin(adata, ["Fcgr1", "Mrc1"], groupby='leiden' )
```

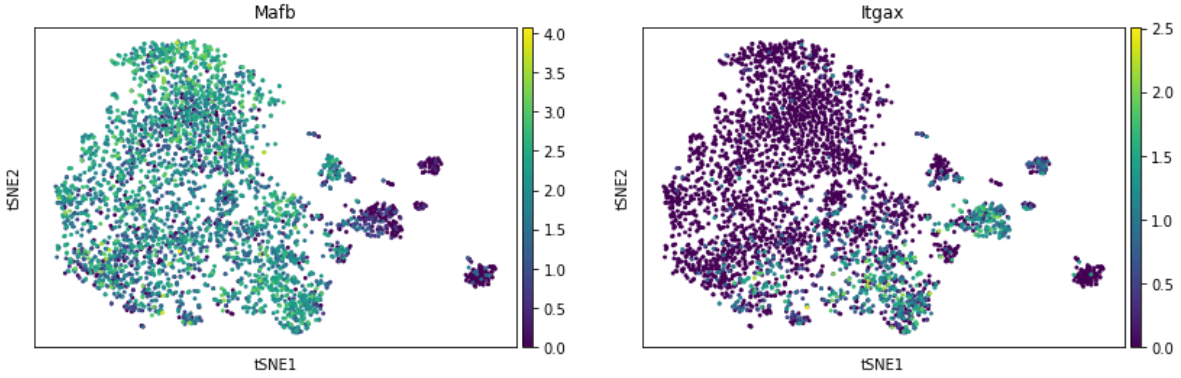


"Mafb", "Itgax" expression



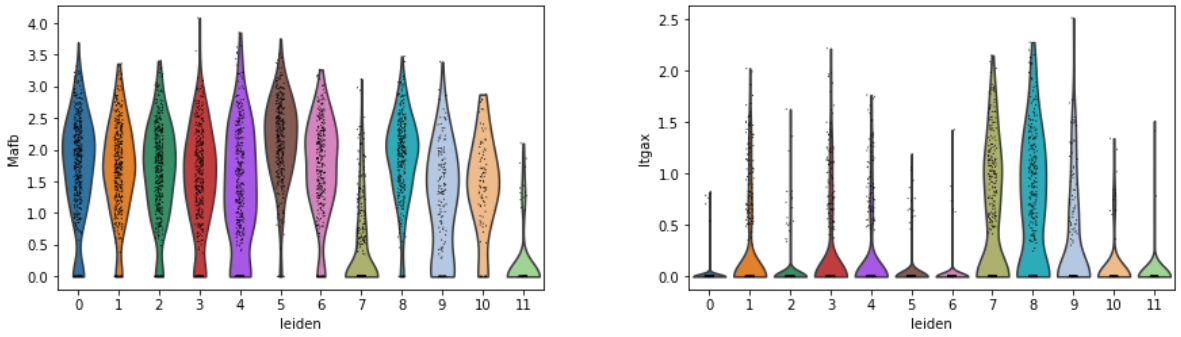
Ввод [28]:

```
sc.pl.tsne(adata, color=["Mafb", "Itgax"])
```

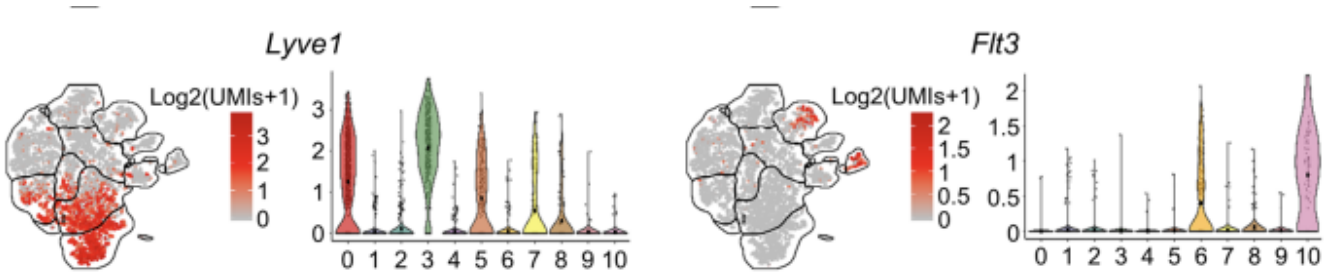


Ввод [29]:

```
sc.pl.violin(adata, ["Mafb", "Itgax"], groupby='leiden' )
```

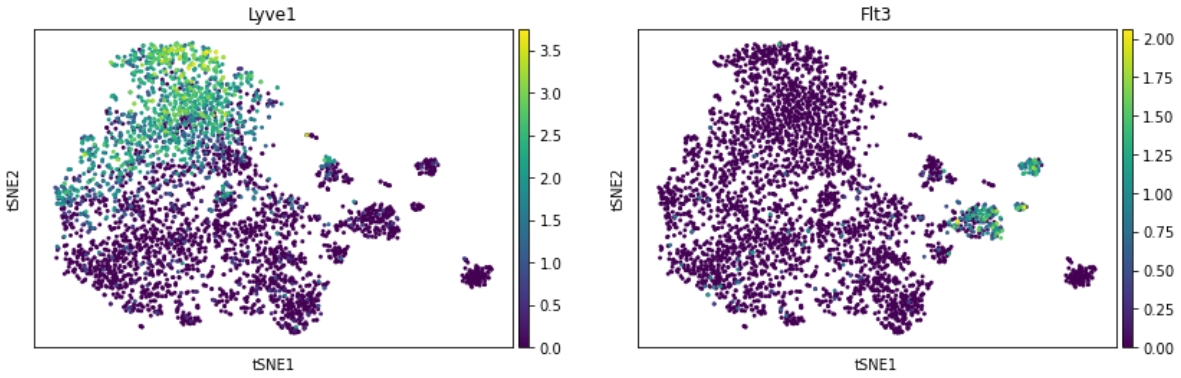


"Lyve1", "Flt3" expression



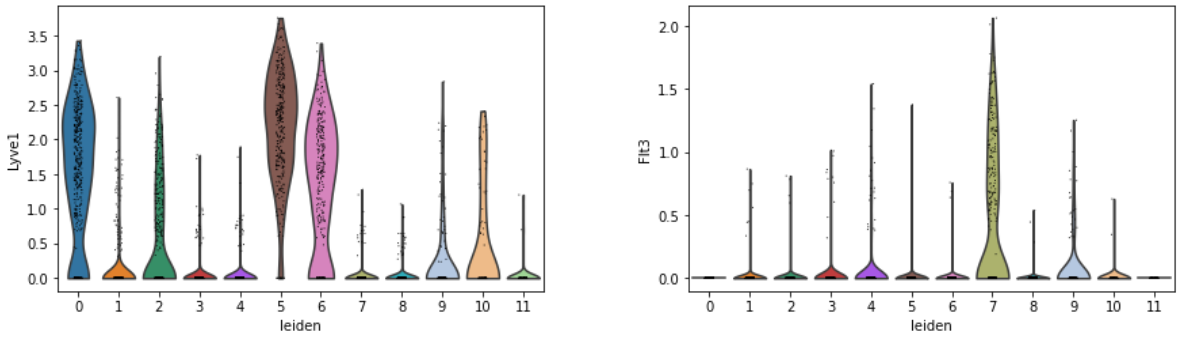
Ввод [30]:

```
sc.pl.tsne(adata, color=["Lyve1", "Flt3"])
```

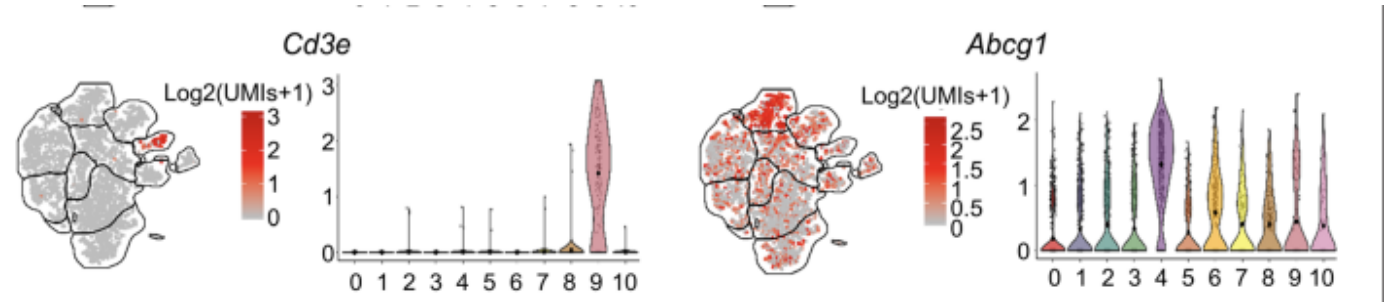


Ввод [31]:

```
sc.pl.violin(adata, ["Lyve1", "Flt3"], groupby='leiden' )
```

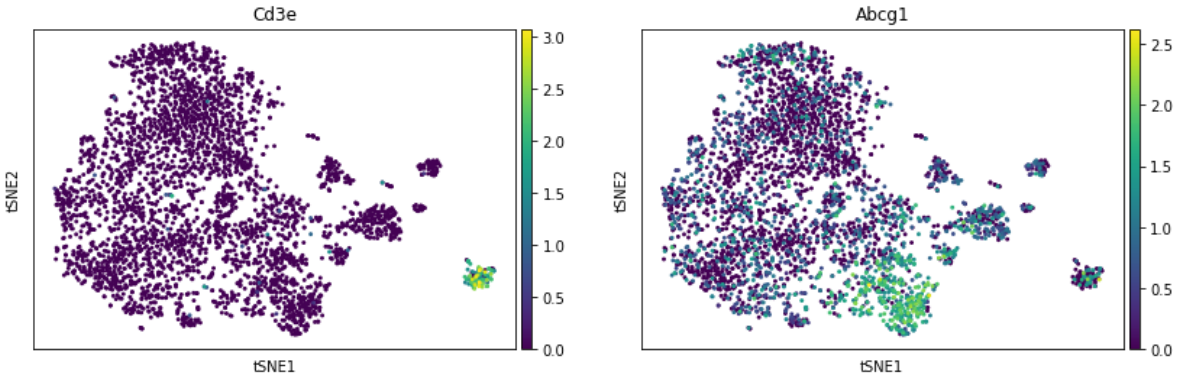


"Cd3e", "Abcg1" expression



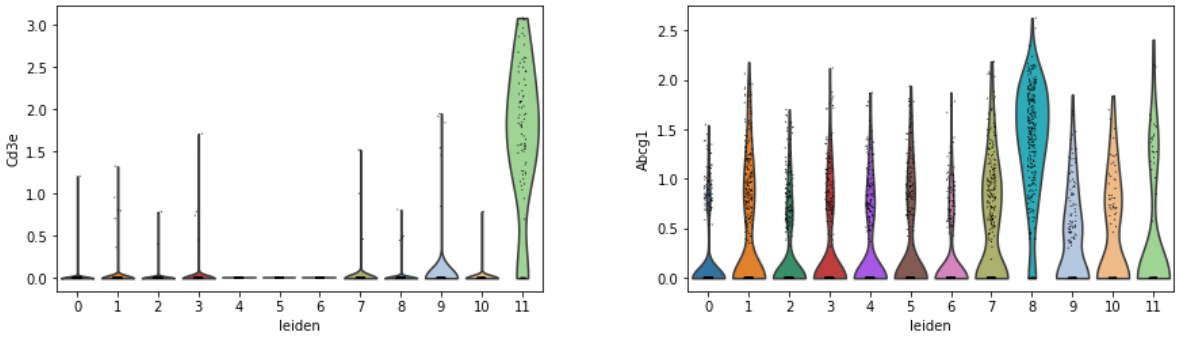
Ввод [32]:

```
sc.pl.tsne(adata, color=["Cd3e", "Abcg1"])
```



Ввод [33]:

```
sc.pl.violin(adata, ["Cd3e", "Abcg1"], groupby='leiden')
```



Expression of principal hematopoietic markers is ditributed in a similar way comparing to the publication results.