CLUSTERING ORGAN CELL TYPES

Project Report

Presented to

Dr. Chris Pollett

Department of Computer Science

San José State University

In Partial Fulfillment

Of the Requirements for the Class

CS 297

By

Swathi MVS

Fall 2023

TABLE OF CONTENTS

I. INTRODUCTION

Every organ in the human body is composed of specific tissues, each tissue is made of specific cells. Cells are the basic building blocks of all living organisms. The human body contains trillions of cells. Each cell has a nucleus which contains DNA - the blueprint of all organisms. The DNA contains genes that carry genetic information. This information has all the instructions for the production of proteins by organisms. The genome is made up of DNA. Our body contains proteins, every protein has a shape and a function. Gene expression represents how much the gene is expressed or activated, which determines the function of a particular protein. The Human Cell Atlas is an international project whose initiative is to create a comprehensive reference map of all human cells. The data for the Human Cell Atlas is obtained from Tabula Sapiens, a project conducted by Chan Zuckerberg Biohub in California. My project takes reference from The Human Cell Atlas and aims to study the Tabula Sapiens Dataset to explore the cells present in various organs and gain insight into the different types of cells present in the human body.

The Human Genome Project (1990-2003) was an international research project and a global effort. It was a 13-year-long project whose goal was to sequence 3 billion base pairs of Human DNA. It also included identifying, mapping, and sequencing all the genes of the human genome from a physical and functional viewpoint. The Human Cell Atlas, similar to the Human Genome Project, is an international collaborative consortium founded in 2016 by Sarah Teichmann and Aviv Regev [1]. The goal is to create a huge map of all the cells in the human body like the human genome did with DNA. The project aims to provide a detailed representation of molecules, cells, tissues, organs, and systems which will allow researchers to identify patterns and interactions at various levels of resolution [2].

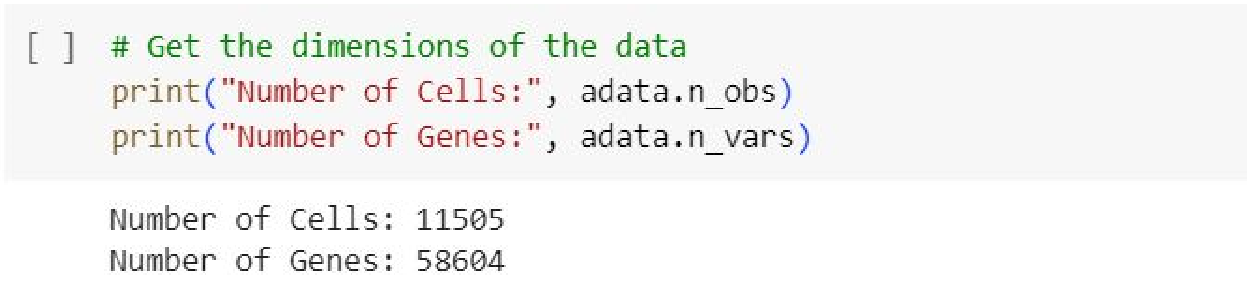
The goal of the project for this semester is to cluster the various types of cells for a particular organ using Machine Learning and be able to distinguish those cell types from others. In the next semester, this research project aims to look into cells of other organs and perform in-depth analysis. The project for this semester is divided into 4 deliverables, each of which is detailed in this report. Deliverable 1 involves researching the human cell data, processing and understanding it. Deliverable 2 deals with implementing various Classification and Clustering techniques on the data. Deliverable 3 focuses on performing Binary Classification on a particular cell type of the data. Deliverable 4 involves building a Neural Network Classifier on the data. Finally, the Conclusion summarises the tasks done this semester and the future goals.

II. DELIVERABLE 1: TABULA SAPIENS DATASET OVERVIEW

This deliverable aims to study and understand the Tabula Sapiens data in order to make use of it in the project. The Tabula Sapiens Dataset is taken from the Human Cell Atlas project. This dataset is a first-draft human cell atlas of nearly 500,000 cells from 24 organs of 15 human donors [3]. It provides insights into the molecular composition of different cell types, containing gene expression patterns, signaling pathways, and so on. It includes data from a broad range of tissues, including the bladder, blood, heart, kidney, mammary, muscle, pancreas, bone marrow, eye, fat, large intestine, liver, lung, lymph node, prostate, salivary gland, skin, and small intestine [4]. The funding for this dataset is done by the Chan Zuckerburg Initiative in San Fransisco.

The Tabula Sapiens data is in H5ad (Hierarchical Data Format 5 Annotated) format. It is based on the standard h5 format i.e. Hierarchical Data Format (HDF) which is used to store scientific data that is well organized for quick retrieval and analysis. This format is specifically used for single-cell RNA sequencing(scRNA-seq) analysis. Annotated data is stored in a hierarchical structure using H5ad files. It is used in combination with AnnData object, part of the Python anndata library. AnnData is a data structure designed to efficiently store and manipulate annotated scRNA-seq data.

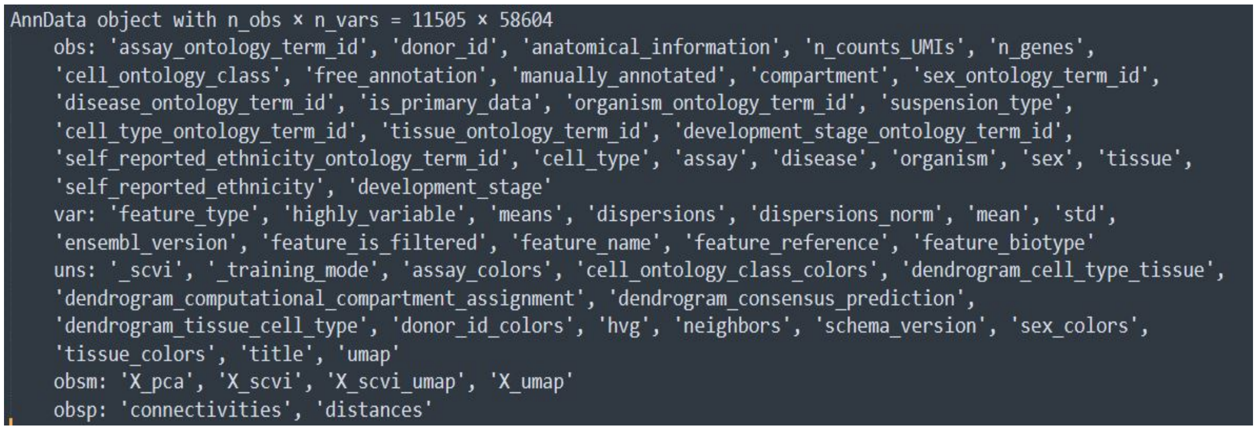
I used scanpy and anndata libraries in Python to read and process the Tabula Sapiens Heart data. The Heart dataset contains 11,505 cells and 58,604 genes which are known as variables and observations respectively.



```
[ ]  # Get the dimensions of the data
     print("Number of Cells:", adata.n_obs)
     print("Number of Genes:", adata.n_vars)

     Number of Cells: 11505
     Number of Genes: 58604
```

Fig 1: Dimensions of Heart data

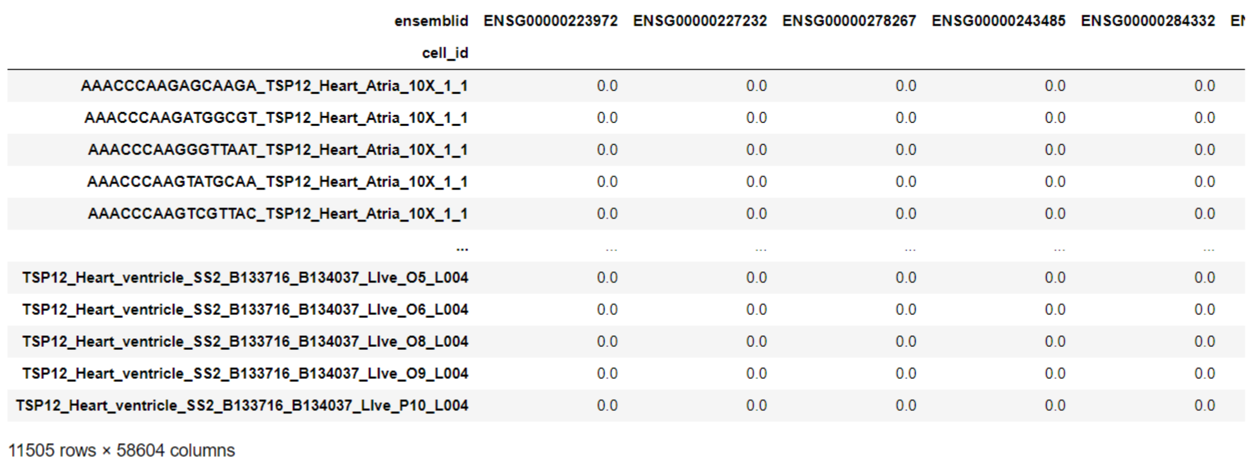
The anndata object of the Heart data contains 5 different attributes, each of which contains sub-attributes. This is shown in Figure 2. The ‘obs’ attribute contains 26 different features that give more information about the cells. ‘donor_id’ represents the id given to the human donor, ‘anatomical_information’ represents where in the heart the cells are present, ‘n_counts_UMI’ represents the count of Unique Molecular Identifier for each cell, ‘n_genes’ represent the number



```
AnnData object with n_obs × n_vars = 11505 × 58604
    obs: 'assay_ontology_term_id', 'donor_id', 'anatomical_information', 'n_counts_UMIs', 'n_genes',
    'cell_ontology_class', 'free_annotation', 'manually_annotated', 'compartment', 'sex_ontology_term_id',
    'disease_ontology_term_id', 'is_primary_data', 'organism_ontology_term_id', 'suspension_type',
    'cell_type_ontology_term_id', 'tissue_ontology_term_id', 'development_stage_ontology_term_id',
    'self_reported_ethnicity_ontology_term_id', 'cell_type', 'assay', 'disease', 'organism', 'sex', 'tissue',
    'self_reported_ethnicity', 'development_stage'
    var: 'feature_type', 'highly_variable', 'means', 'dispersions', 'dispersions_norm', 'mean', 'std',
    'ensembl_version', 'feature_is_filtered', 'feature_name', 'feature_reference', 'feature_biotype'
    uns: '_scvi', '_training_mode', 'assay_colors', 'cell_ontology_class_colors', 'dendrogram_cell_type_tissue',
    'dendrogram_computational_compartment_assignment', 'dendrogram_consensus_prediction',
    'dendrogram_tissue_cell_type', 'donor_id_colors', 'hvg', 'neighbors', 'schema_version', 'sex_colors',
    'tissue_colors', 'title', 'umap'
    obsm: 'X_pca', 'X_scvi', 'X_scvi_umap', 'X_umap'
    obsp: 'connectivities', 'distances'
```

Figure 2: Heart Data attributes

of genes present in each cell, ‘cell_ontology_class’ represents the high-level cell classes, ‘manually_annotated’ represents whether the cell is manually labeled, ‘cell_type’ represents the type of heart cell, and so on. The ‘var’ attribute contains 12 features which give details about the genes. ‘Feature_type’ represents the gene expression, ‘highly_variable’ represents whether the gene is highly variable, ‘means’, and ‘dispersions’ represent the mean and dispersion of the gene expression, ‘feature_name’ represents the name of the gene, and so on. ‘uns’ represents the unstructured annotations. ‘obsm’ represents the multidimensional observations annotation of length number of annotations [5]. ‘obsp’ represents the properties of the cells. In Figure 3, the rows of the data frame ‘cell_id’ represent the Unique Molecular Identifier of a cell, and columns ‘ensemblid’ represent the gene identifier of the genes. Each row represents how much a particular gene is expressed or active in a particular cell.



ensemblid cell_id	ENSG00000223972	ENSG00000227232	ENSG00000278267	ENSG00000243485	ENSG00000284332
AAACCCAAGAGCAAGA_TSP12_Heart_Atria_10X_1_1	0.0	0.0	0.0	0.0	0.0
AAACCCAAGATGGCGT_TSP12_Heart_Atria_10X_1_1	0.0	0.0	0.0	0.0	0.0
AAACCCAAGGGTTAAT_TSP12_Heart_Atria_10X_1_1	0.0	0.0	0.0	0.0	0.0
AAACCCAAGTATGCAA_TSP12_Heart_Atria_10X_1_1	0.0	0.0	0.0	0.0	0.0
AAACCCAAGTCGTTAC_TSP12_Heart_Atria_10X_1_1	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...
TSP12_Heart_ventricle_SS2_B133716_B134037_LIve_O5_L004	0.0	0.0	0.0	0.0	0.0
TSP12_Heart_ventricle_SS2_B133716_B134037_LIve_O6_L004	0.0	0.0	0.0	0.0	0.0
TSP12_Heart_ventricle_SS2_B133716_B134037_LIve_O8_L004	0.0	0.0	0.0	0.0	0.0
TSP12_Heart_ventricle_SS2_B133716_B134037_LIve_O9_L004	0.0	0.0	0.0	0.0	0.0
TSP12_Heart_ventricle_SS2_B133716_B134037_LIve_P10_L004	0.0	0.0	0.0	0.0	0.0

11505 rows × 58604 columns

Figure 3: Heart Dataframe

I performed dimensionality reduction of Heart data using Principal Component Analysis (PCA) and Uniform Manifold Approximation and Projection (UMAP). Figure 4 and 5 show PCA and UMAP plots of Heart data.

PCA is a common dimensionality reduction technique to reduce the dimensionality of huge datasets. PCA increases the interpretability of the data while preserving the maximum amount of information, and enables the visualization of multidimensional data. It takes all the input features and combines them to produce new features called 'principal components'. These principal components are correlated with each other and are ranked from the most important to the least important. PCA works by standardizing the continuous input features and computes the covariance matrix to identify correlations among the features. Then the eigenvectors and eigenvalues of the covariance matrix are calculated to determine the principal components. These principal components are the input features projected onto a unit vector to preserve the information. The eigenvector with the largest eigenvalue signifies the most important principal component [6].

In Figure 4, PC1 constitutes the first principal component, the line that best represents the data in lower dimensional space and has maximum variance in this direction. PC2 constitutes the next significant principal component, which best represents the data.

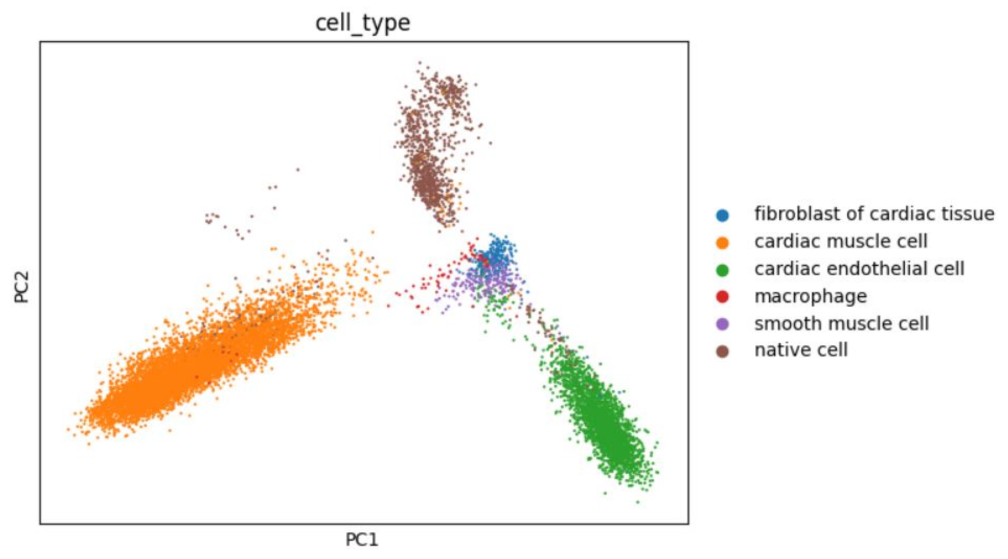


Figure 4: PCA of Heart data

UMAP is a new and fast dimensionality reduction method based on neighbour graphs. UMAP constructs a high-dimensional graph representation of the actual data and then maps it to a low-dimensional graph to be as structurally similar to it as possible. The shape of the high-dimensional data can be approximated by connecting the data points called 0-simplices with their neighbouring data points forming 1 or 2 or higher dimensional simplices [7]. UMAP algorithm uses a radius around each data point to make a connection between each data point and its neighbours with intersecting radii. This forms a connected graph of all the data points.

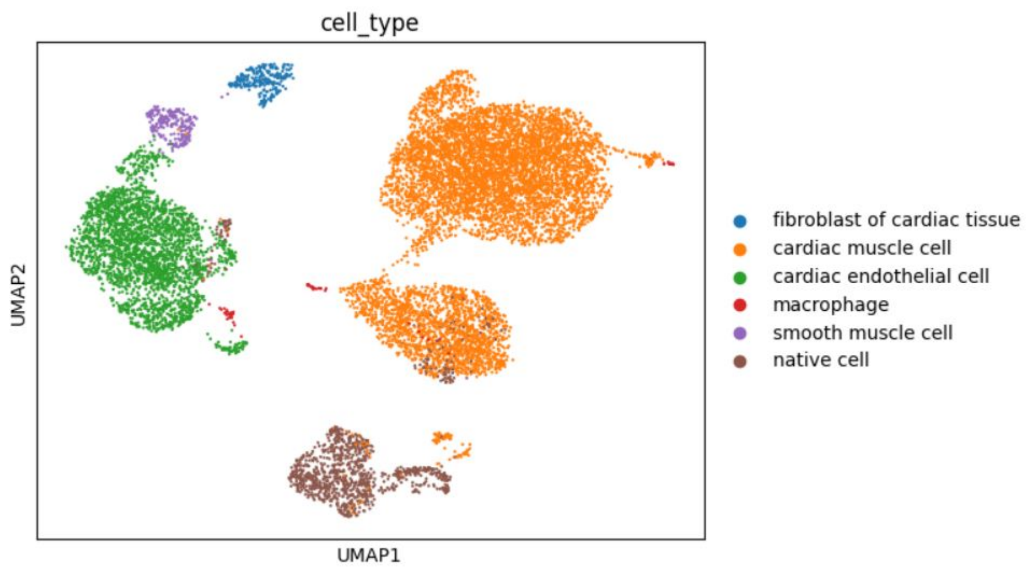


Figure 5: UMAP of Heart data

Choosing the radius is crucial and UMAP chooses the radius locally based on the distance to each point's nth nearest neighbour [8]. This ensures that both the local and global structures are preserved. The connection between each point and its neighbours has a weight, this is the connection probability. Points that are farther away weigh less and closer points weigh more. Points connected by high-weighted edges are more likely to stay together in low-dimensional space. Once the graph is connected, it is projected to a lower dimension using a graph projection algorithm. Overall, UMAP is fast and has a better balance of local and global structuring.

In Figure 5, the X and Y axes represent a UMAP space where the distances in the two-dimensional space match the high-dimensional space of the original data as closely as possible.

III. DELIVERABLE II: CLASSIFICATION AND CLUSTERING OF HEART DATA

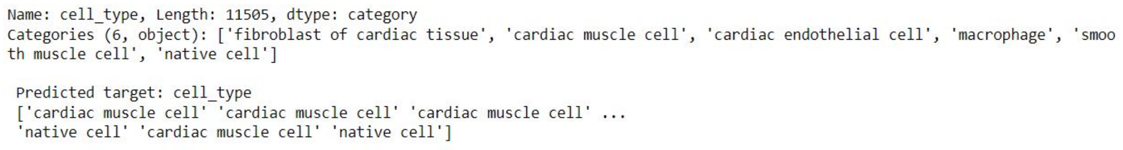
This deliverable involves implementing Classification and Clustering algorithms on the Heart data.

1) Classification

   The objective of classification is to assign pre-defined classes or labels to instances. The purpose is to predict the class or label of unseen instances. It is a supervised learning technique. It requires labeled data for training. The output is either a class or label assignment. Example: Predicting whether an email is spam or not.

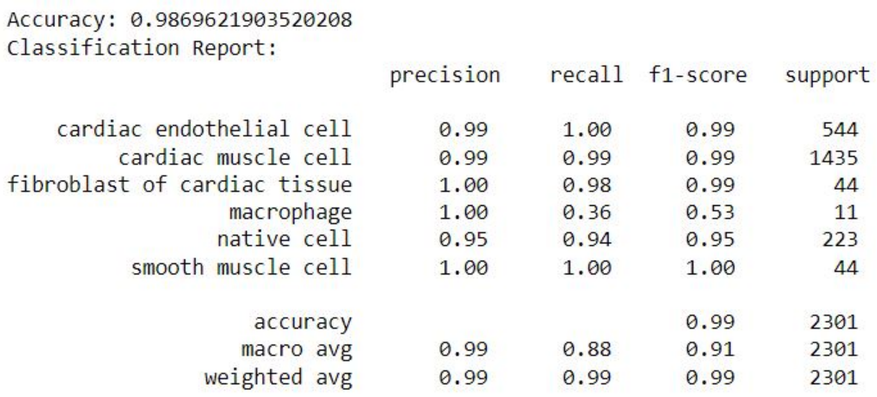
   a) Logistic Regression

      It is a linear classification algorithm used for binary and multiclass classification tasks. It models the probability of an instance belonging to a particular class. I performed logistic regression on the Heart data to predict the cell type by splitting 80% as training data and 20% as testing data with maximum iterations of 1000. Figure 6 shows the predicted cell type.



```
Name: cell_type, Length: 11505, dtype: category
Categories (6, object): ['fibroblast of cardiac tissue', 'cardiac muscle cell', 'cardiac endothelial cell', 'macrophage', 'smoo
th muscle cell', 'native cell']

 Predicted target: cell_type
 ['cardiac muscle cell' 'cardiac muscle cell' 'cardiac muscle cell' ...
 'native cell' 'cardiac muscle cell' 'native cell']
```

Figure 6: Logistic Regression predicted cell type



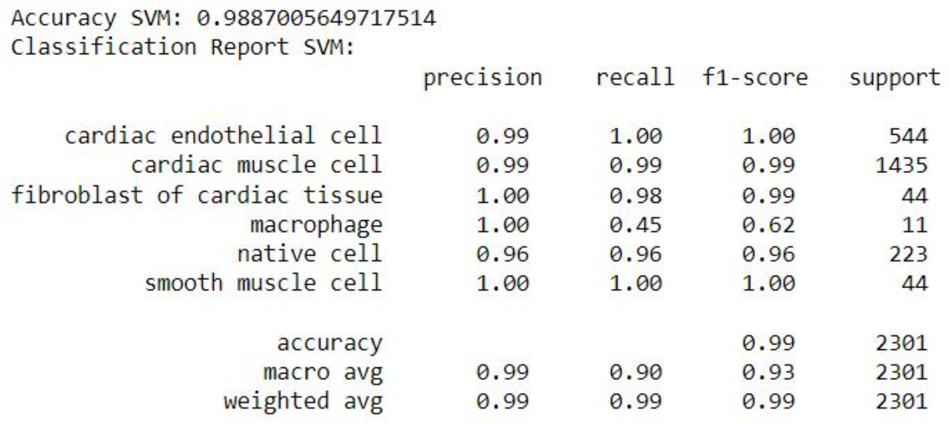
```
Accuracy: 0.9869621903520208
Classification Report:
                              precision    recall  f1-score   support

    cardiac endothelial cell       0.99      1.00      0.99       544
         cardiac muscle cell       0.99      0.99      0.99      1435
fibroblast of cardiac tissue       1.00      0.98      0.99        44
                  macrophage       1.00      0.36      0.53        11
                 native cell       0.95      0.94      0.95       223
           smooth muscle cell      1.00      1.00      1.00        44

                    accuracy                           0.99      2301
                   macro avg       0.99      0.88      0.91      2301
                weighted avg       0.99      0.99      0.99      2301
```

Figure 7: Logistic Regression accuracy on cell type

Figure 7 shows logistic regression accuracy on cell type. The model gave fairly good results with an accuracy of 98.6%.

b) Support Vector Machine (SVM)

SVMs are used for classification tasks and aim to find a hyperplane that maximizes the margin between classes. I implemented SVM on Heart data to predict the cell type by splitting 80% as training data and 20% as testing data with a linear kernel and C as 1.0. Figure 8 shows the SVM accuracy on cell type. The accuracy of SVM is pretty good at 98.8% which is slightly better than logistic regression.



```
Accuracy SVM: 0.9887005649717514
Classification Report SVM:
                              precision    recall  f1-score   support

    cardiac endothelial cell       0.99      1.00      1.00       544
         cardiac muscle cell       0.99      0.99      0.99      1435
fibroblast of cardiac tissue       1.00      0.98      0.99        44
                  macrophage       1.00      0.45      0.62        11
                 native cell       0.96      0.96      0.96       223
           smooth muscle cell       1.00      1.00      1.00        44

                    accuracy                           0.99      2301
                   macro avg       0.99      0.90      0.93      2301
                weighted avg       0.99      0.99      0.99      2301
```

Figure 8: SVM accuracy on cell type

2) Clustering

The objective of clustering is to group similar instances based on similarity. The purpose is to discover inherent patterns or structures in the data. It is an unsupervised learning technique. It does not require labeled data for training. The output is cluster assignment. Example: Grouping customers based on purchasing behaviour.

a) K-Means Clustering

K-means is one of the most popular clustering algorithms. It partitions data into K clusters, with each data point assigned to the nearest cluster center. I performed K-Means clustering on Heart data. Figure 9 shows the different clusters.

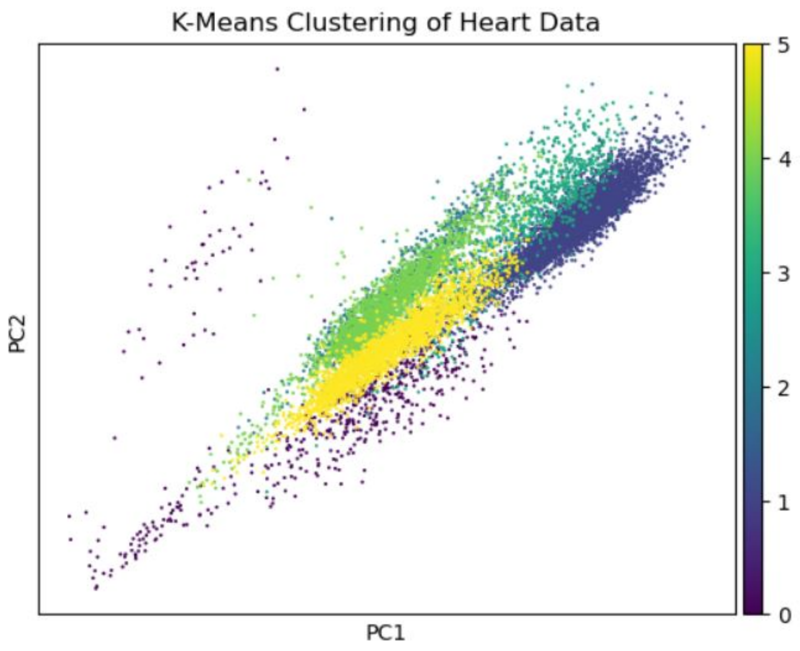


Figure 9: K-means on Heart data

Figure 10 shows the different cluster centers. Most of the cell types were grouped to different clusters.

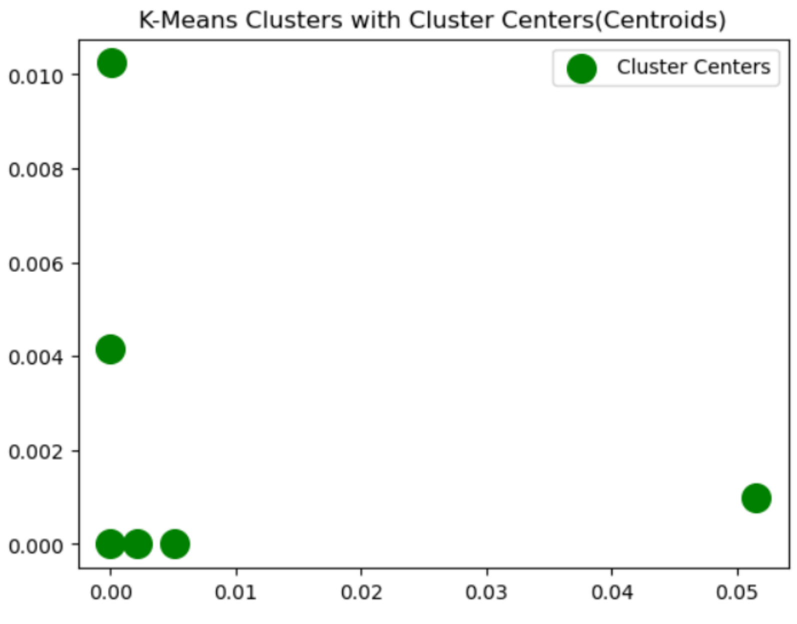


Figure 10: K-means cluster centers

b) Hierarchical Clustering

Hierarchical clustering builds a tree-like structure of clusters, either bottom-up (agglomerative) or top-down (divisive), allowing for different levels of granularity in cluster assignments. I implemented hierarchical clustering on Heart data with a max_distance of 2000. Figure 11 shows the output of hierarchical clustering. The clusters aren't clear from the plotted principal components.

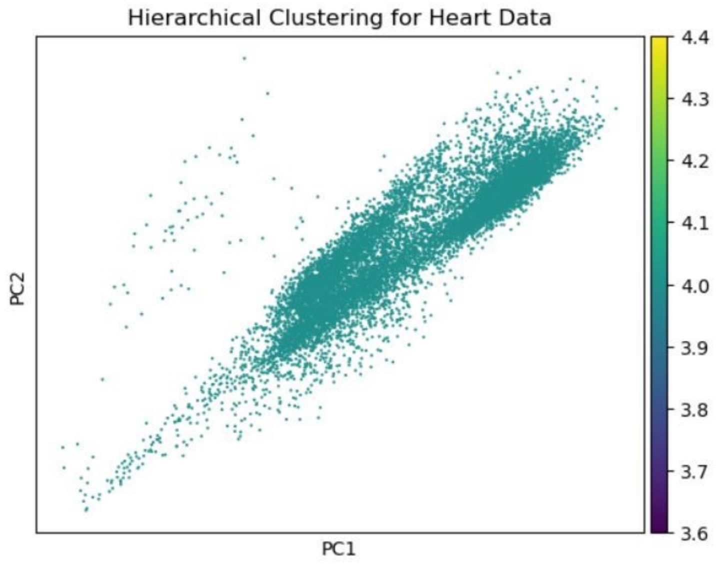


Figure 11: Hierarchical clustering on cell type

Figure 12 shows the dendrogram of hierarchical clustering.

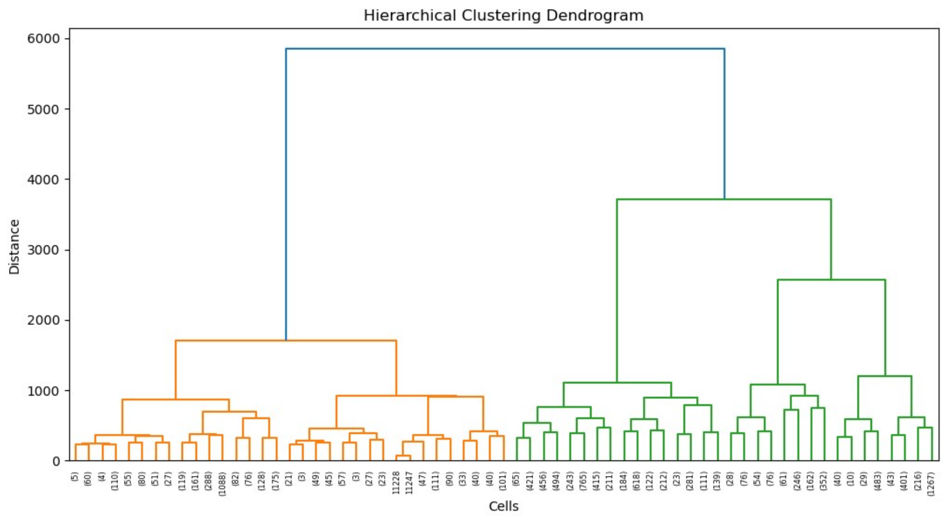


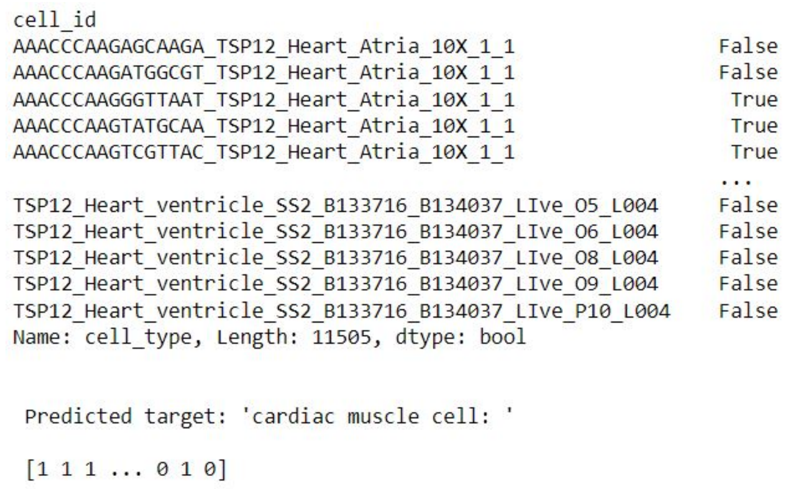
Figure 12: Hierarchical clustering dendrogram

IV. DELIVERABLE III: BINARY CLASSIFICATION ON A CELL TYPE OF HEART DATA

The aim of this deliverable is to select a cell type from Heart data and perform Binary Classification on it to check how well it can be distinguished from other cell types.

a) Logistic Regression

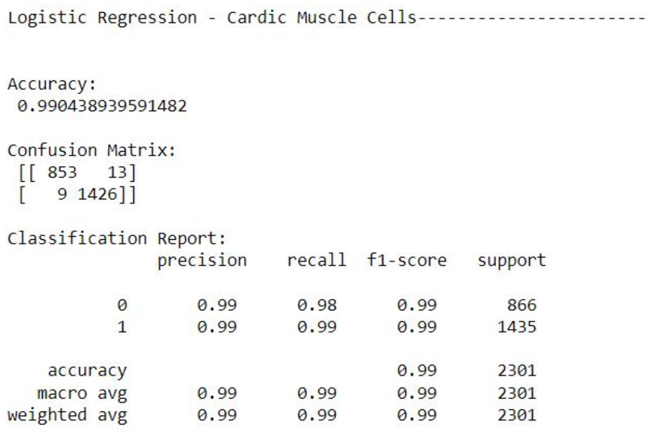
I chose cardiac muscle cells cell type of Heart data to perform logistic regression. Implemented logistic regression by adding a binary label for cardiac muscle cells and splitting the data as 80% training data and 20% testing data with maximum iterations of 1000. Figure 12 shows the predicted output.



```
cell_id
AAACCCAAGAGCAAGA_TSP12_Heart_Atria_10X_1_1                False
AAACCCAAGATGGCGT_TSP12_Heart_Atria_10X_1_1                False
AAACCCAAGGGTTAAT_TSP12_Heart_Atria_10X_1_1                 True
AAACCCAAGTATGCAA_TSP12_Heart_Atria_10X_1_1                 True
AAACCCAAGTCGTTAC_TSP12_Heart_Atria_10X_1_1                 True
                                                          ...
TSP12_Heart_ventricle_SS2_B133716_B134037_LIve_O5_L004    False
TSP12_Heart_ventricle_SS2_B133716_B134037_LIve_O6_L004    False
TSP12_Heart_ventricle_SS2_B133716_B134037_LIve_O8_L004    False
TSP12_Heart_ventricle_SS2_B133716_B134037_LIve_O9_L004    False
TSP12_Heart_ventricle_SS2_B133716_B134037_LIve_P10_L004   False
Name: cell_type, Length: 11505, dtype: bool


 Predicted target: 'cardiac muscle cell: '

 [1 1 1 ... 0 1 0]
```

Figure 12: Logistic regression on cardiac muscle cells

Figure 13 shows an accuracy of 99.04% which is very good.

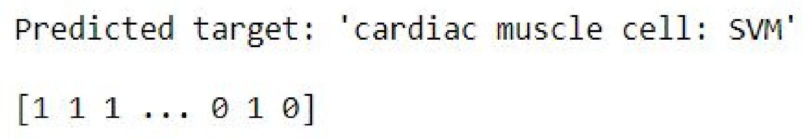


```
Logistic Regression - Cardic Muscle Cells-----------------------


Accuracy:
 0.990438939591482

Confusion Matrix:
 [[ 853   13]
 [   9 1426]]

Classification Report:
               precision    recall  f1-score   support

           0       0.99      0.98      0.99       866
           1       0.99      0.99      0.99      1435

    accuracy                           0.99      2301
   macro avg       0.99      0.99      0.99      2301
weighted avg       0.99      0.99      0.99      2301
```

Figure 13: Logistic regression accuracy on cardiac muscle cells

b) Support Vector Machine

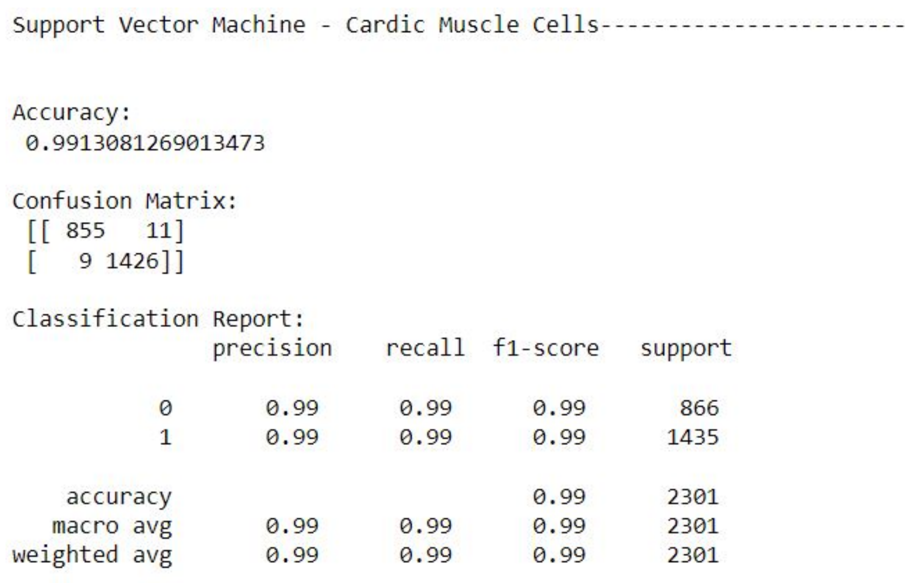
I chose cardiac muscle cells cell type of Heart data to implement SVM. Implemented SVM by adding a binary label for cardiac muscle cells and splitting the data as 80% training data and 20% testing data with linear kernel and C as 1. Figure 14 shows the predicted output.



```
Predicted target: 'cardiac muscle cell: SVM'

[1 1 1 ... 0 1 0]
```

Figure 14: SVM on cardiac muscle cells

Figure 15 shows the accuracy of SVM on cardiac muscle cells. The accuracy of 99.13% is impressive which is slightly higher than logistic regression accuracy.



```
Support Vector Machine - Cardic Muscle Cells-----------------------


Accuracy:
 0.9913081269013473

Confusion Matrix:
 [[ 855   11]
 [   9 1426]]

Classification Report:
               precision    recall  f1-score   support

           0       0.99      0.99      0.99       866
           1       0.99      0.99      0.99      1435

    accuracy                           0.99      2301
   macro avg       0.99      0.99      0.99      2301
weighted avg       0.99      0.99      0.99      2301
```

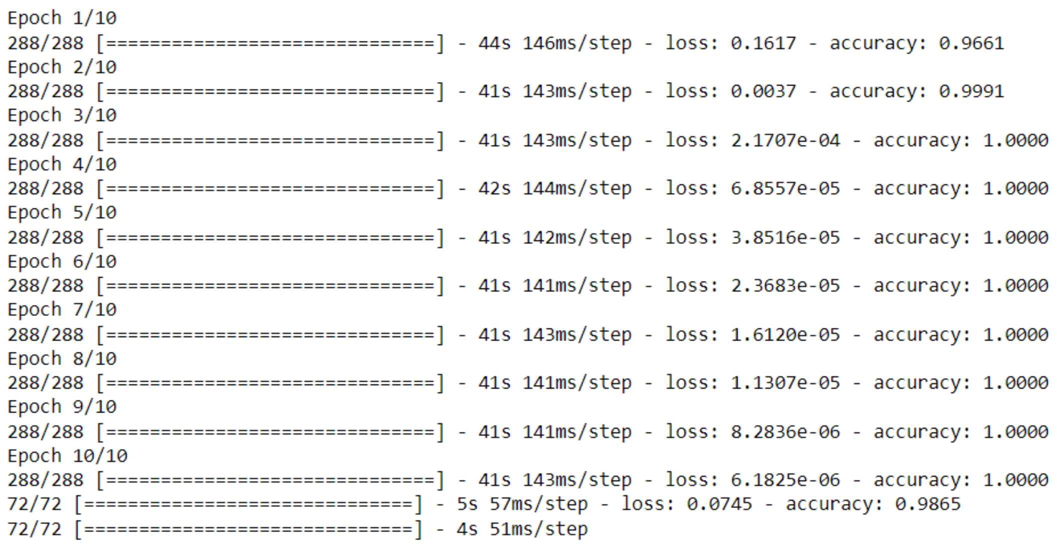
Figure 15: SVM accuracy on cardiac muscle cells

V. DELIVERABLE IV: NEURAL NETWORK CLASSIFICATION

This deliverable aims to build a Neural Network Classifier for the Heart data. Neural Networks try to replicate the brain. They take inspiration from learning process that occurs in human brains [9]. They are made up of an artificial network of functions known as parameters that the computer uses to analyze fresh data in order to learn and adjust. Every parameter, sometimes known as neurons, is a function that, given one or more inputs, generates an output. The subsequent layer of neurons receives those outputs and uses them as inputs for their own function, producing more outputs. The following layer of neurons receives those outputs, and so on until all layers of neurons have been taken into account and the terminal neurons have been fed. The model's ultimate output is subsequently produced by those terminal neurons.

I implemented the Neural Network Classification on Heart data. The architecture of the Neural Network is as follows. The input features had to be converted to floating point data and the categorical output cell type had to be label encoded to integers. It consists of 2 hidden layers. The input layer receives raw data from the Heart dataset. It consists of 128 neurons and takes input as the shape of training data, these act as a buffer, receiving the raw data and passing it on to the next layer for further processing. The second layer consists of 64 neurons, responsible for transforming the input data in a non-linear manner. This layer introduces non-linearity into the network, enabling it to capture complex relationships between the input features. The activation function used in this layer is the ReLU (Rectified Linear Unit) function. ReLU outputs the input if it is positive and zero otherwise, effectively introducing a non-linear threshold to the data. This non-linearity allows the network to learn more complex patterns in the data, which is crucial for accurate cell type classification. The output layer consists of six neurons, one for each cell type in the dataset. Each neuron in this layer represents the probability that the input belongs to the corresponding cell type. The activation function used in this layer is the softmax function. Softmax normalizes the outputs of the neurons so that they sum to one, ensuring that the probabilities for all cell types are mutually exclusive. This makes the output suitable for interpreting as probabilities, where each neuron represents the likelihood that the input belongs to the corresponding cell type.

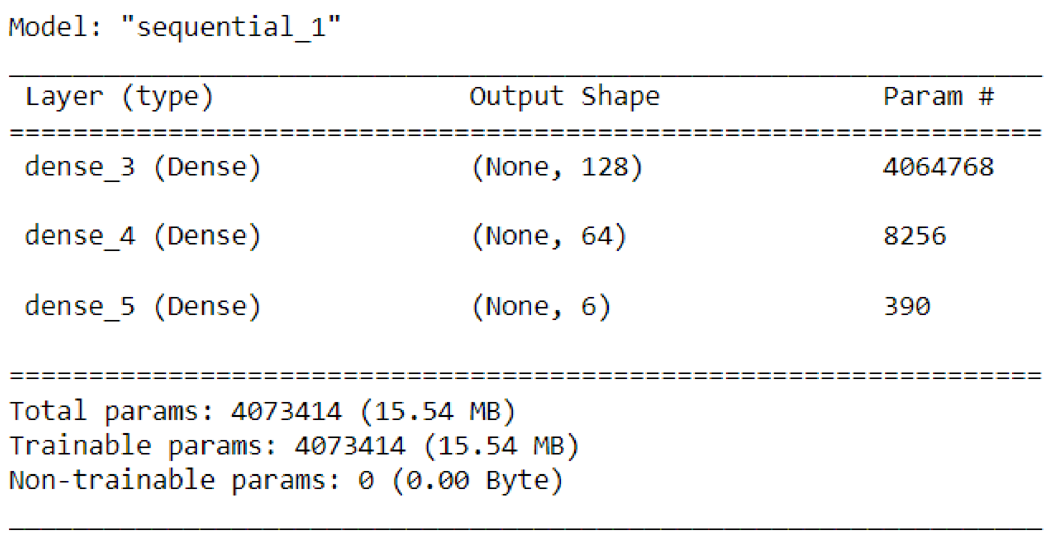
The sparse categorical cross-entropy loss function is used to measure the difference between the predicted cell types and the actual cell types in the training data. This loss function is appropriate for multi-class classification tasks, where each input belongs to one of several mutually exclusive categories. As the model trains, it iteratively updates the weights between the neurons in order to minimize this loss function. By minimizing the loss, the model learns to correctly classify the cell types in the training data. The Adam optimizer is used to efficiently update the weights between neurons during training. Adam is a popular choice for training neural networks due to its ability to handle noisy data and its ability to converge quickly to a good solution. It effectively adjusts the weights in a direction that reduces the loss, allowing the model to learn the patterns in the data and improve its classification accuracy. The model is run for 10 epochs. Accuracy is used as a metric to evaluate the model's performance.



```
Epoch 1/10
288/288 [==============================] - 44s 146ms/step - loss: 0.1617 - accuracy: 0.9661
Epoch 2/10
288/288 [==============================] - 41s 143ms/step - loss: 0.0037 - accuracy: 0.9991
Epoch 3/10
288/288 [==============================] - 41s 143ms/step - loss: 2.1707e-04 - accuracy: 1.0000
Epoch 4/10
288/288 [==============================] - 42s 144ms/step - loss: 6.8557e-05 - accuracy: 1.0000
Epoch 5/10
288/288 [==============================] - 41s 142ms/step - loss: 3.8516e-05 - accuracy: 1.0000
Epoch 6/10
288/288 [==============================] - 41s 141ms/step - loss: 2.3683e-05 - accuracy: 1.0000
Epoch 7/10
288/288 [==============================] - 41s 143ms/step - loss: 1.6120e-05 - accuracy: 1.0000
Epoch 8/10
288/288 [==============================] - 41s 141ms/step - loss: 1.1307e-05 - accuracy: 1.0000
Epoch 9/10
288/288 [==============================] - 41s 141ms/step - loss: 8.2836e-06 - accuracy: 1.0000
Epoch 10/10
288/288 [==============================] - 41s 143ms/step - loss: 6.1825e-06 - accuracy: 1.0000
72/72 [==============================] - 5s 57ms/step - loss: 0.0745 - accuracy: 0.9865
72/72 [==============================] - 4s 51ms/step
```

Figure 16: Neural Network accuracy

Figure 16 shows the accuracy of the neural network model. It has a high accuracy of 98.65%.



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_3 (Dense)             (None, 128)               4064768

 dense_4 (Dense)             (None, 64)                8256

 dense_5 (Dense)             (None, 6)                 390

=================================================================
Total params: 4073414 (15.54 MB)
Trainable params: 4073414 (15.54 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 17: Neural Network model summary

Figure 17 shows the summary of the Neural Network model.

VI. CONCLUSION

The purpose of this project is to analyze the Tabula Sapiens Heart data and implement clustering and classification mechanisms to identify the cell types in the Heart data. Each of the four deliverables has contributed to this objective. In the future, this research aims to be extended to various other organs and cell types present in the Tabula Sapiens dataset for further analysis.

Deliverable 1 focused on exploring the Tabula Sapiens Heart data, understanding the format, analyzing the features, what the data contains, and processing the data. This was specific to Heart cells and gave an overview of what the data represents, which may be carried out on all the different cells present in the Tabula Sapiens data in the next semester. Deliverable 2 included the implementation of a few Classification and Clustering algorithms on Heart data to understand how well the data can be classified into cell types and if it forms good clusters. Moving forward, clustering algorithms can be emphasized more. Deliverable 3 involved selecting a particular cell type from the Heart data and performing Binary Classification on the selected cell type to distinguish whether it can be separately classified from other cell types. Deliverable 4 employed Deep Learning by building a Neural Network model to classify the Heart Data. This concludes the project for this semester.

The goal of this project is to group the distinct cell types available in the Tabula Sapiens dataset into clusters to see if we can discover unknown cell types. The project for the next semester focuses on implementing various clustering techniques based on Neural Networks on the Tabula Sapiens data to identify any new cell types. It also involves exploring Neural Network embeddings to cluster the data.

REFERENCES

[1] A Cartography of Human Histology in the making
https://www.economist.com/science-and-technology/2023/03/08/a-cartography-of-human-histology-is-in-the-making

[2] The Human Cell Atlas: from vision to reality https://www.nature.com/articles/550451a

[3] Tabula Sapiens Dataset https://tabula-sapiens-portal.ds.czbiohub.org/

[4] The Tabula Sapiens Consortium, The Tabula Sapiens: A multiple-organ, single-cell transcriptomic atlas of humans.Science376,eabl4896(2022).DOI:10.1126/science.abl4896

[5] Anndata Documentation
https://anndata.readthedocs.io/en/latest/generated/anndata.AnnData.html

[6] A Step-by-Step Explanation of Principal Component Analysis(PCA)
https://builtin.com/data-science/step-step-explanation-principal-component-analysis

[7] UMAP explained | The best dimensionality reduction?
https://www.youtube.com/watch?v=6BPl81wGGP8

[8] Understanding UMAP
https://pair-code.github.io/understanding-umap/

[9] Classification using Neural Networks
https://towardsdatascience.com/classification-using-neural-networks-b8e98f3a904f