Module_3:

Team Members:

Hazel Miranda and Michael Dornic

Project Title:

Understanding Gene Dysregulation in the Hallmark "Evading Apoptosis" Across Cancer Types

Project Goal:

This project seeks to analyze how cancer cells evade apoptosis by examining the expression and mutation patterns of key apoptotic genes on Breast Cancer.

Disease Background

- *Cancer hallmark focus:*

  Evading Apoptosis (Resisting Cell Death)

- *Overview of hallmark:*

  *Apoptosis, or programmed cell death, is a tightly controlled process that removes damaged or unnecessary cells. In healthy breast tissue, apoptosis maintains balance during development, lactation, and involution. However, in breast cancer, tumor cells often acquire mutations that disable apoptotic signaling, allowing them to survive beyond their normal lifespan. The tumor suppressor p53 (encoded by *TP53*) normally senses DNA damage and triggers apoptosis, but when mutated, it fails to activate death pathways. In addition, the BCL-2 family of proteins regulates mitochondrial membrane permeability—the critical checkpoint for intrinsic apoptosis. Overexpression of anti-apoptotic proteins (like *BCL2*) or loss of pro-apoptotic proteins (*BAX*, *BAK*) blocks apoptosis and enables tumor progression.* *Understanding how breast cancer cells selectively manipulate these pathways can reveal why some tumors respond to apoptosis-targeting therapies while others resist them.*

  cite: https://www.sciencedirect.com/science/article/pii/S0092867411001279?via%3Dihub

cite: https://blog.cellsignal.com/hallmarks-of-cancer-resisting-cell-death?

- *Genes associated with hallmark to be studied:*

  | **Gene** | **Function in Apoptosis** | **Role in Breast Cancer** | | --- | --- | --- | | **TP53** | Tumor suppressor that detects DNA damage and triggers apoptosis through activation of BAX/BAK and caspases | Mutated in ~30–40 % of breast cancers; loss of function prevents apoptosis and promotes therapy resistance | | **BCL2** | Anti-apoptotic protein that prevents mitochondrial outer-membrane permeabilization | Overexpressed in many ER-positive breast cancers; high expression linked to cell survival and chemotherapy resistance | | **BAX** | Pro-apoptotic effector that promotes cytochrome c release from mitochondria | Downregulated in invasive ductal carcinoma; imbalance between BCL-2 and BAX shifts cells toward survival | | **BAK** | Works with BAX to form pores in mitochondrial membranes and initiate caspase activation | Reduced activity in aggressive subtypes limits intrinsic apoptosis | | **CASP8** | Initiator caspase in the extrinsic (death receptor) pathway | Sometimes silenced in breast tumors, blocking death-receptor–mediated apoptosis |

  Cite: https://www.gsea-msigdb.org/gsea/msigdb/human/geneset/HALLMARK_APOPTOSIS.html?

  Cite: https://www.ncbi.nlm.nih.gov/books/NBK26884/

- *Will you be focusing on a single cancer type or looking across cancer types?*

  We will focus on a single cancer type: breast cancer (BRCA). The TCGA Breast Cancer dataset (RNA-Seq and mutation data for > 1,000 tumor samples) allows detailed analysis of apoptosis-related genes.

  Subtypes show distinct apoptotic profiles:

  - *Luminal A/B (ER-positive): High BCL2, lower BAX*
  - *HER2-positive: Often amplifies MCL1 and overexpresses BCL2*
  - *Triple-negative (TNBC): Frequently loses *TP53* function and downregulates BAK/BAX**

  Cite: https://www.cancer.gov/ccg/research/genome-sequencing/tcga

- *Prevalence & incidence:*

  - *Breast cancer is the most common cancer among women worldwide, with ~2.3 million new cases and ~680,000 deaths annually (WHO, 2024).*
  - *In the United States, an estimated 4,091,181 women were living with Breast cancer in 2022.

    - *Approximately 131 new cases of Breast cancer are reported for every 100,000 women every year (NIH).
    - *HER2-positive tumors account for 15–20 % of cases and are typically more aggressive and apoptosis-resistant.*
    - *Triple-negative cancers have the poorest prognosis due to limited targeted therapies and frequent TP53 loss.*

  Cite: https://www.who.int/news-room/fact-sheets/detail/cancer https://seer.cancer.gov/statfacts/html/breast.html
- *Risk factors (genetic, lifestyle) & societal determinants:*
    - *Genetic: BRCA1/2 and TP53 mutations; overexpression of BCL2 and MCL1 linked to therapy resistance.*
    - *Lifestyle: Alcohol use, obesity, lack of exercise, and long-term hormone therapy increase risk.*
    - *Societal: Access to early screening and treatment greatly improves outcomes; healthcare inequities lead to later diagnosis and higher mortality.*
    - *Age: Most diagnozed cases are in women aged 50 or older, and the risk increases with age.

  Cite: https://www.cancer.gov/types/breast/risk-factors https://www.cdc.gov/breast-cancer/risk-factors/index.html

- *Standard of care treatments (& reimbursement):*
    - *Surgery, radiation, and chemotherapy** remain standard first-line therapies.*
    - *Targeted therapies:*
        - *Trastuzumab (Herceptin) and Pertuzumab block HER2 signaling and can restore apoptosis.*
        - *Venetoclax (BCL-2 inhibitor) is under study for ER-positive breast cancer.*
    - *Reimbursement: Traditional therapies are covered by most insurers; newer targeted agents may face high costs and limited coverage.*

  Cite: https://www.fda.gov/drugs/resources-information-approved-drugs/venclexta-venetoclax

  Cite: https://www.nature.com/articles/nrc.2018.7

- *Biological mechanisms (anatomy, organ physiology, cell & molecular physiology):*
    - *Anatomy: Originates from epithelial cells lining breast ducts and lobules.*
    - *Organ physiology: Apoptosis is essential for breast tissue remodeling during puberty, pregnancy, and weaning.*
    - *Cell & molecular physiology:*

    - *Intrinsic pathway: DNA damage -> p53 activation -> BAX/BAK -> cytochrome c release -> caspase cascade -> apoptosis.*
    - *Extrinsic pathway: Death receptor binding -> CASP8 activation -> executioner caspases.*
    - *In breast cancer, overexpression of BCL2 or loss of TP53/BAX/BAK prevents cell death even after DNA damage or chemotherapy.*

  Cite: https://www.ncbi.nlm.nih.gov/books/NBK26884/

Data-Set:

- *Data-set*

For this project, we are using data derived from The Cancer Genome Atlas (TCGA), which includes gene expression and clinical information across multiple cancer types. Our analysis focuses specifically on breast cancer (TCGA-BRCA) and the hallmark Evading Apoptosis (Resisting Cell Death).

The data used comes from the GSE62944 TCGA Pan-Cancer RNA-Seq dataset (Rahman et al., Bioinformatics, 2015) and the TCGA Pan-Cancer Clinical Data Resource (TCGA-CDR). Both have been curated and subsampled for educational use in this module

Cite: use of cleaning data set

- *Dataset Description:*
    - *Expression data:*

  File: GSE62944_subsample_log2TPM.csv

  Contains RNA-Seq expression values (log_2-transformed TPM) for the most variable protein-coding genes across TCGA cancer types.

  We selected only breast cancer (BRCA) samples for our analysis.

  Expression data were generated using the Rsubread alignment and quantification pipeline on Illumina HiSeq RNA-Seq platforms.

  Expression levels are measured in Transcripts Per Million (TPM), which were log_2-transformed (log2(TPM + 1)) for normalization and variance stabilization.

Cite: use of cleaning data set

- *Dataset Description:*
    - *Expression data:*

  File: GSE62944_metadata.csv

Includes demographic, tumor stage, survival, and molecular subtype information for each patient sample (≈80 variables after curation).

Metadata are linked to expression data by TCGA sample barcodes, allowing integration of gene expression with clinical outcomes.

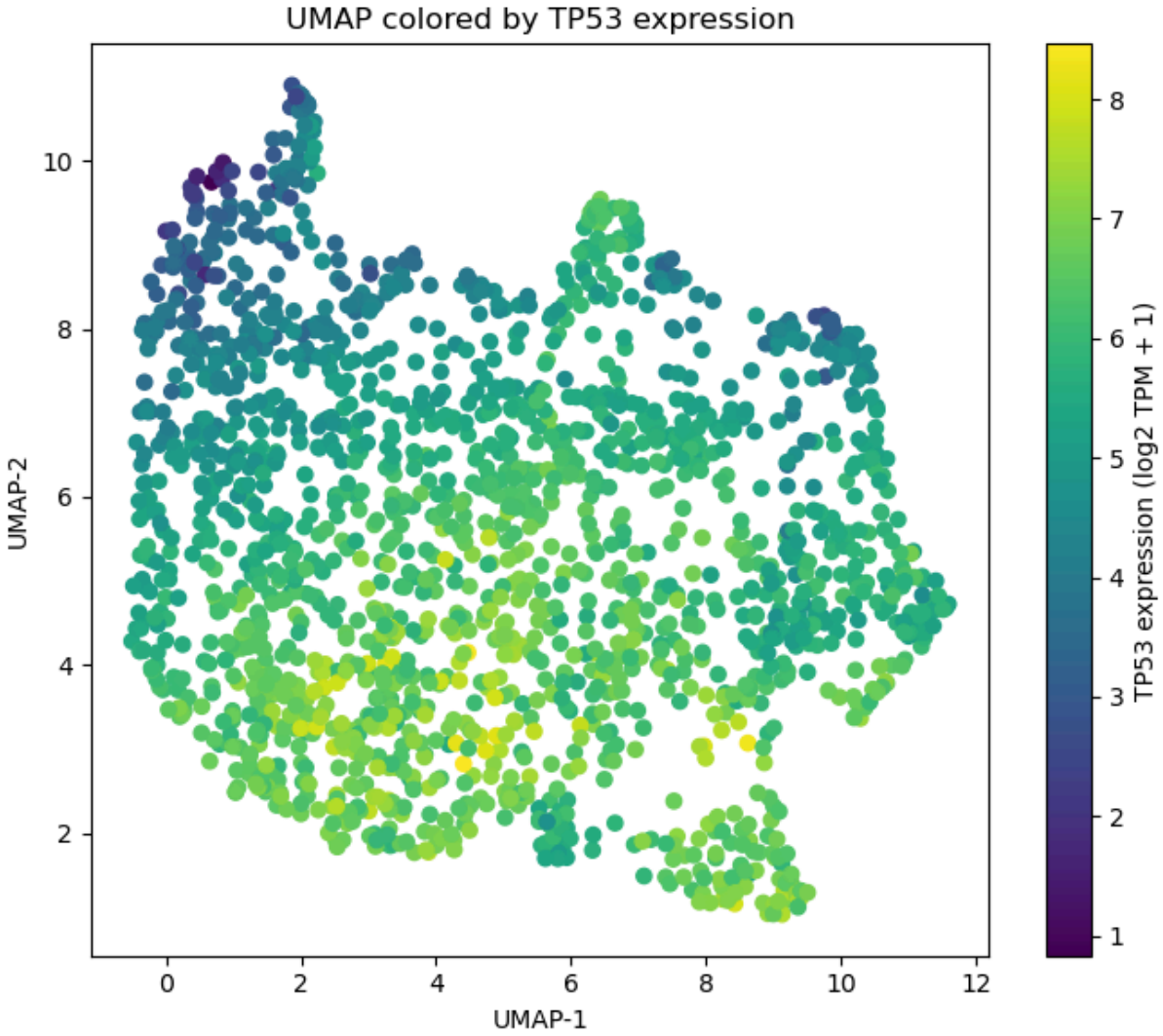
Data include important features such as age, gender, tumor stage (AJCC/TNM), receptor status (ER/PR/HER2), and survival times from the TCGA-CDR dataset.

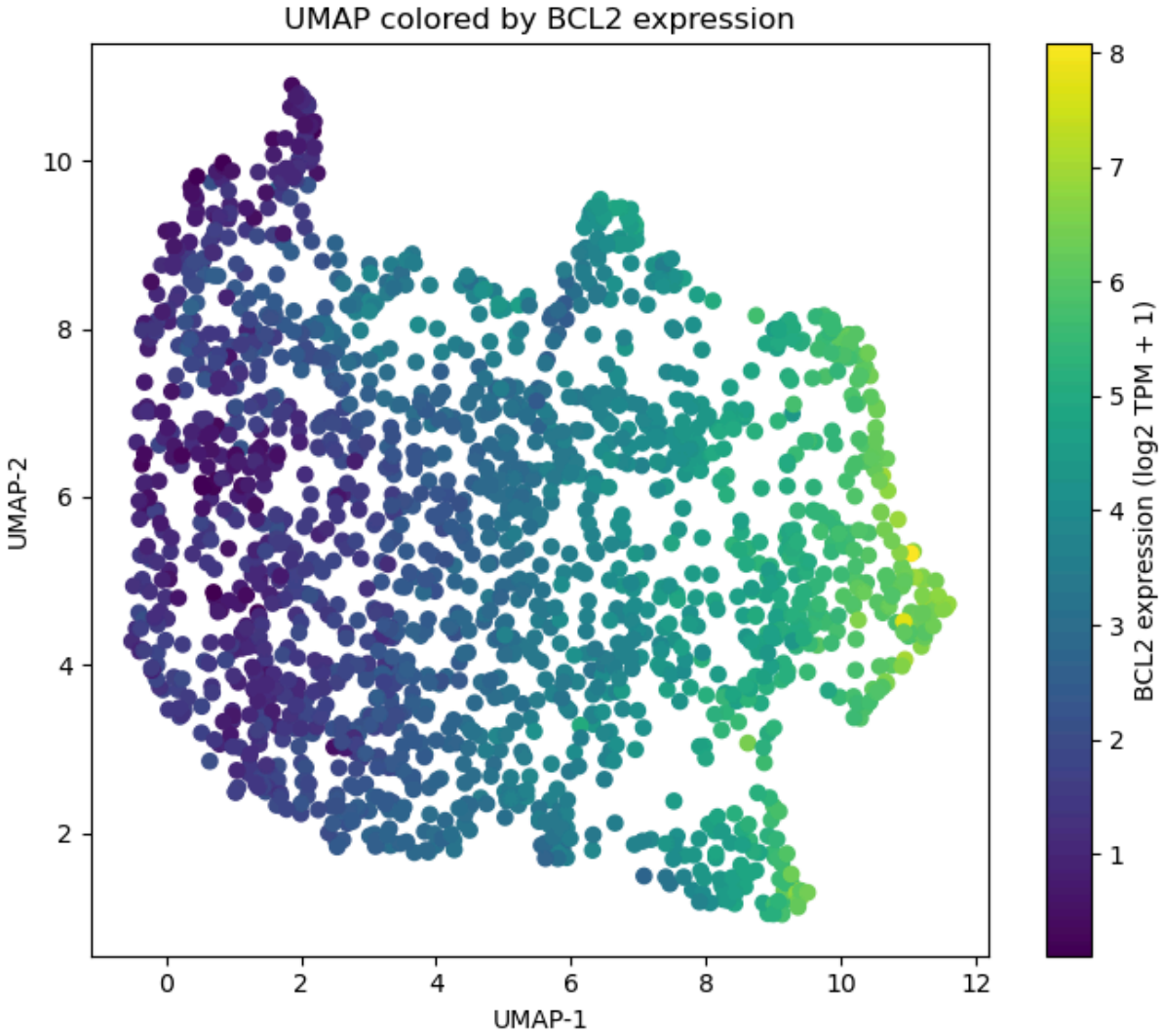
- *Subset used for this project:*
  - *Cancer type: Breast Invasive Carcinoma (BRCA)*
  - *Genes of interest: TP53, BCL2, BAX, and BAK*
  - *These genes were chosen based on their roles in regulating apoptosis:*
    - *TP53 activates BAX and BAK to trigger mitochondrial apoptosis.*
    - *BCL2 inhibits this process by blocking cytochrome c release.*
    - *Clinical features analyzed: molecular subtype, survival status, and stage.*
- *Data Collection & Processing Techniques:*
  - *RNA-Seq data were originally generated by TCGA and reprocessed by Rahman et al. using the Rsubread pipeline to ensure consistent alignment and quantification across 9,264 tumor samples from 24 cancer types.*
  - *Only protein-coding genes (as defined by GENCODE v19) were retained, and low-expression genes were filtered out (TPM ≥ 1 in at least 10% of samples)*
  - *Data were log_2-transformed and stratified across cancer types to maintain balanced sample representation.*
  - *The breast cancer subset was then extracted for focused analysis of apoptotic regulation.*

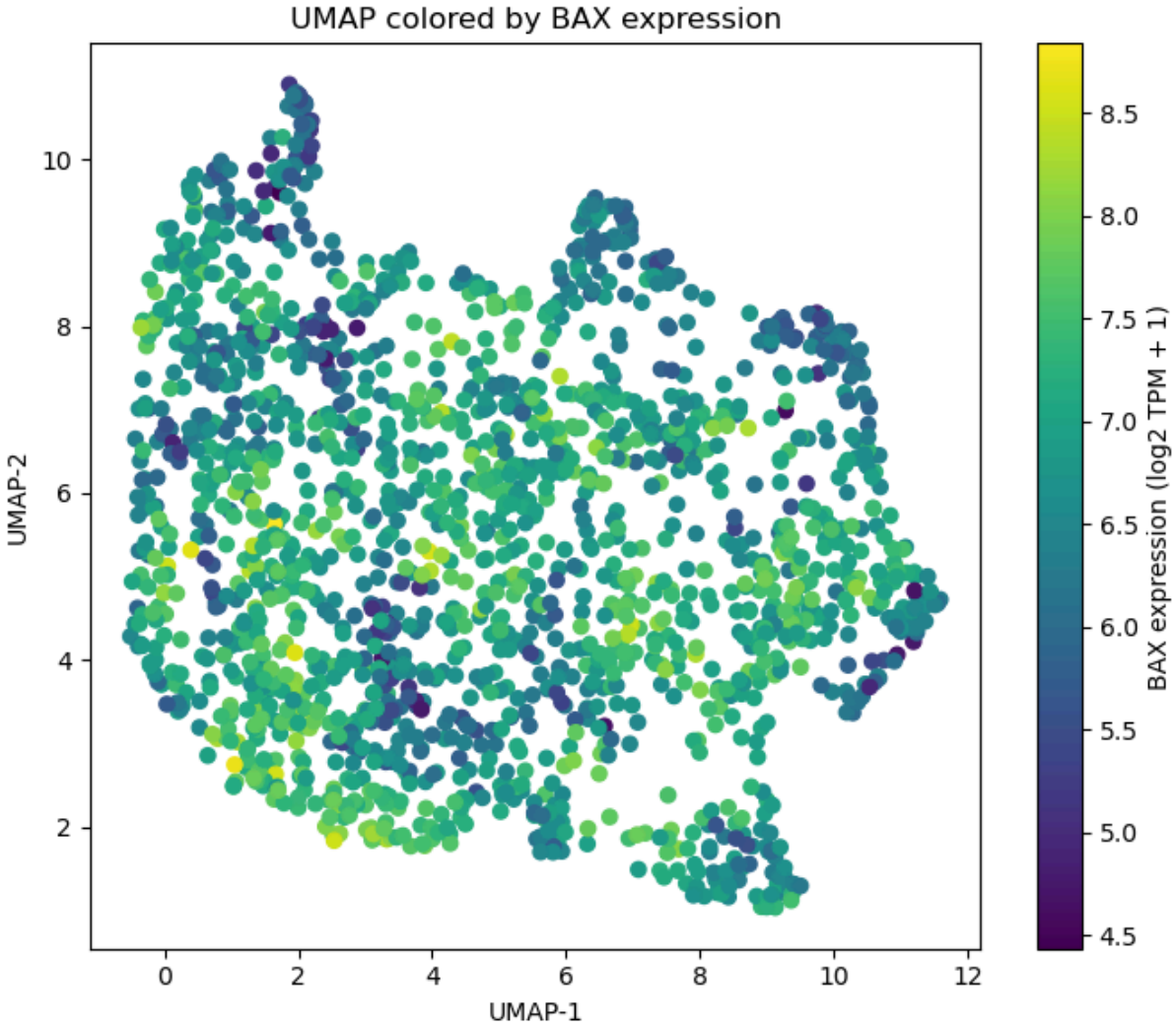
  Cite: use of cleaning data
- *Units & Format:*
  - *Gene expression: $log_2(TPM + 1)$*
  - *Clinical data: categorical (e.g., ER/PR/HER2 status), numerical (e.g., age, survival time in days), and ordinal (e.g., AJCC stage).*
  - *Data structure: genes × samples matrix (expression) and sample × features table (metadata).*

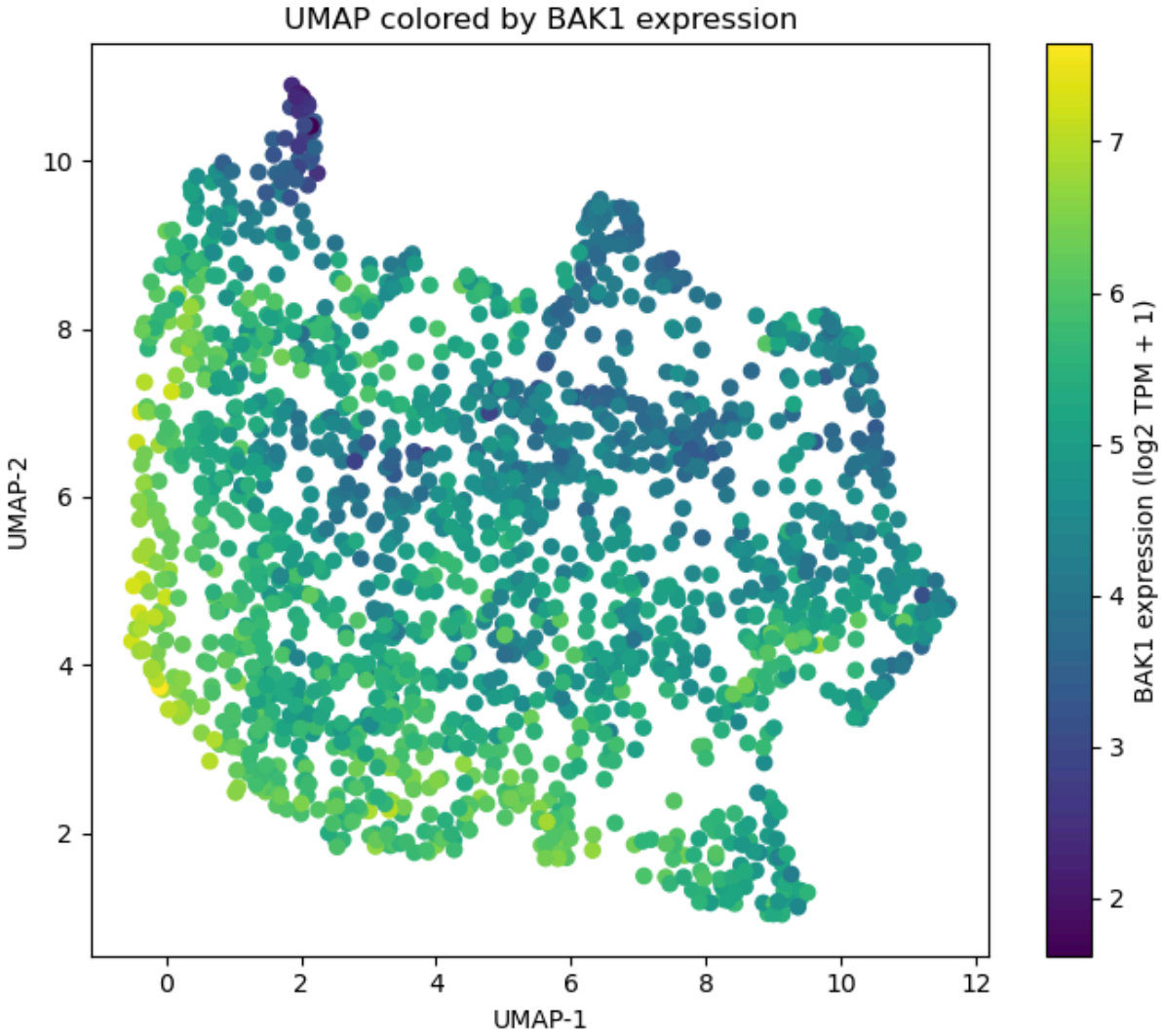
In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import umap
import numpy as np

# --- Load data (note that you might have to rename file name) ---
expr = pd.read_csv("/Users/mikedornic/Desktop/bruh/UVA/BME/GSE62944_subsampl
meta = pd.read_csv("/Users/mikedornic/Desktop/bruh/UVA/BME/GSE62944_metadata

# --- Transpose (samples as rows, genes as columns) ---
expr = expr.T
expr.columns = expr.columns.astype(str).str.split('|').str[0].str.upper()

# --- Genes of interest ---
genes = ["TP53", "BCL2", "BAX", "BAK1", "CASP8"]

present = [g for g in genes if g in expr.columns]
print("Genes found:", present)
if len(present) < 2:
    raise ValueError("Need at least two genes for UMAP.")

# --- Align sample IDs ---
common = expr.index.intersection(meta.index)
expr, meta = expr.loc[common], meta.loc[common]

# --- UMAP embedding (using all apoptosis genes together) ---
X = expr[present]
reducer = umap.UMAP(n_neighbors=15, min_dist=0.3, random_state=42)
embedding = reducer.fit_transform(X)

# --- Make a color map for each gene ---
for gene in present:
    plt.figure(figsize=(7,6))
    plt.scatter(embedding[:,0], embedding[:,1],
                c=X[gene], cmap="viridis", s=35)
    plt.colorbar(label=f"{gene} expression (log2 TPM + 1)")
    plt.xlabel("UMAP-1")
    plt.ylabel("UMAP-2")
    plt.title(f"UMAP colored by {gene} expression")
    plt.tight_layout()
    plt.show()
```

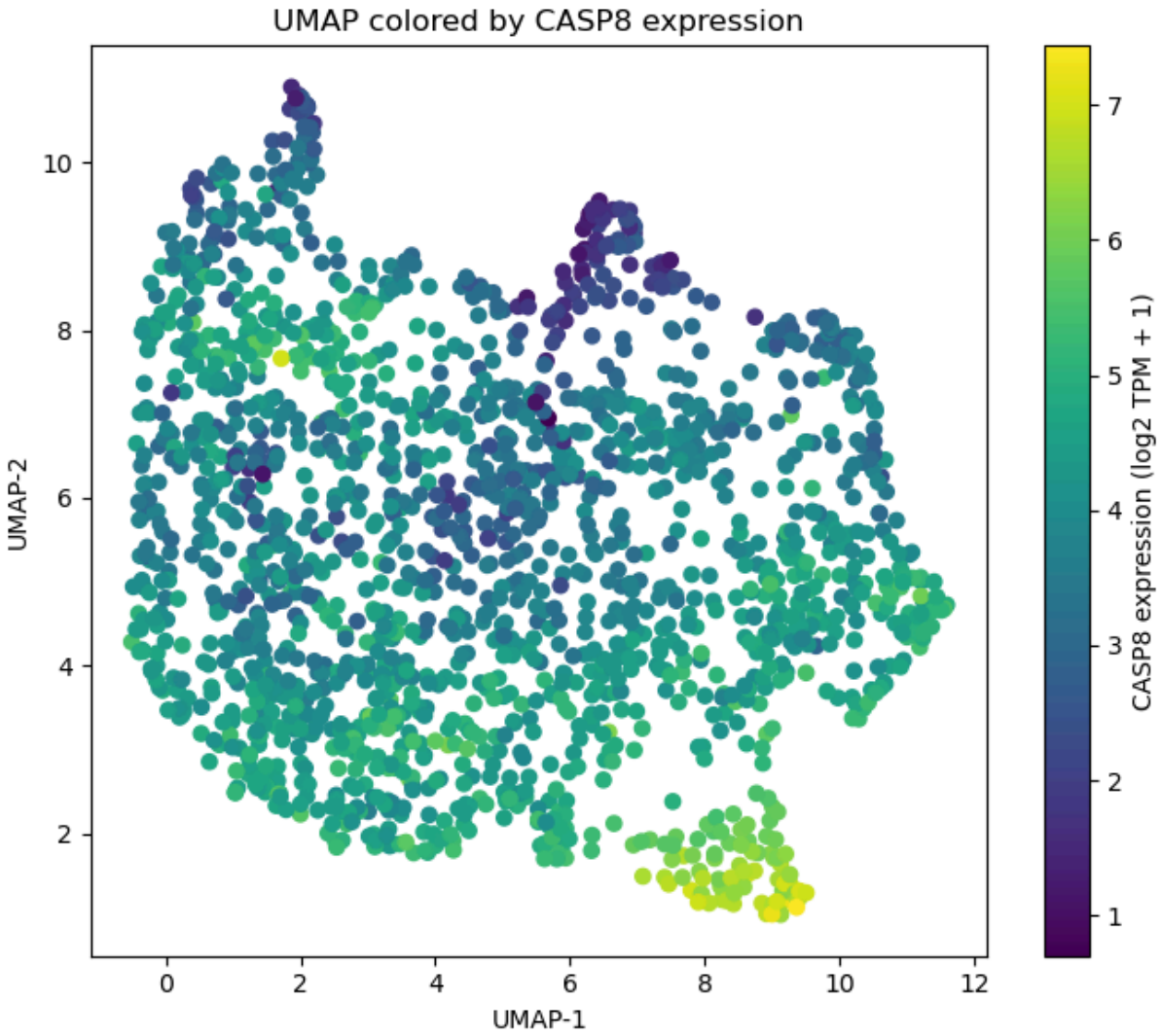
```
Genes found: ['TP53', 'BCL2', 'BAX', 'BAK1', 'CASP8']
/opt/anaconda3/lib/python3.13/site-packages/umap/umap_.py:1952: UserWarning:
n_jobs value 1 overridden to 1 by setting random_state. Use no seed for para
llelism.
  warn(
```


UMAP colored by TP53 expression
UMAP-2
UMAP-1
TP53 expression (log2 TPM + 1)
0
2
4
6
8
10
12
1
2
3
4
5
6
7
8


UMAP colored by BCL2 expression
UMAP-2
UMAP-1
BCL2 expression (log2 TPM + 1)
0
2
4
6
8
10
12
1
2
3
4
5
6
7
8


UMAP colored by BAX expression
UMAP-1
UMAP-2
BAX expression (log2 TPM + 1)
0
2
4
6
8
10
12
2
4
6
8
10
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
8.5


UMAP colored by BAK1 expression
UMAP-2
UMAP-1
BAK1 expression (log2 TPM + 1)
0
2
4
6
8
10
12
2
3
4
5
6
7



Code With Test/Split

In [4]:

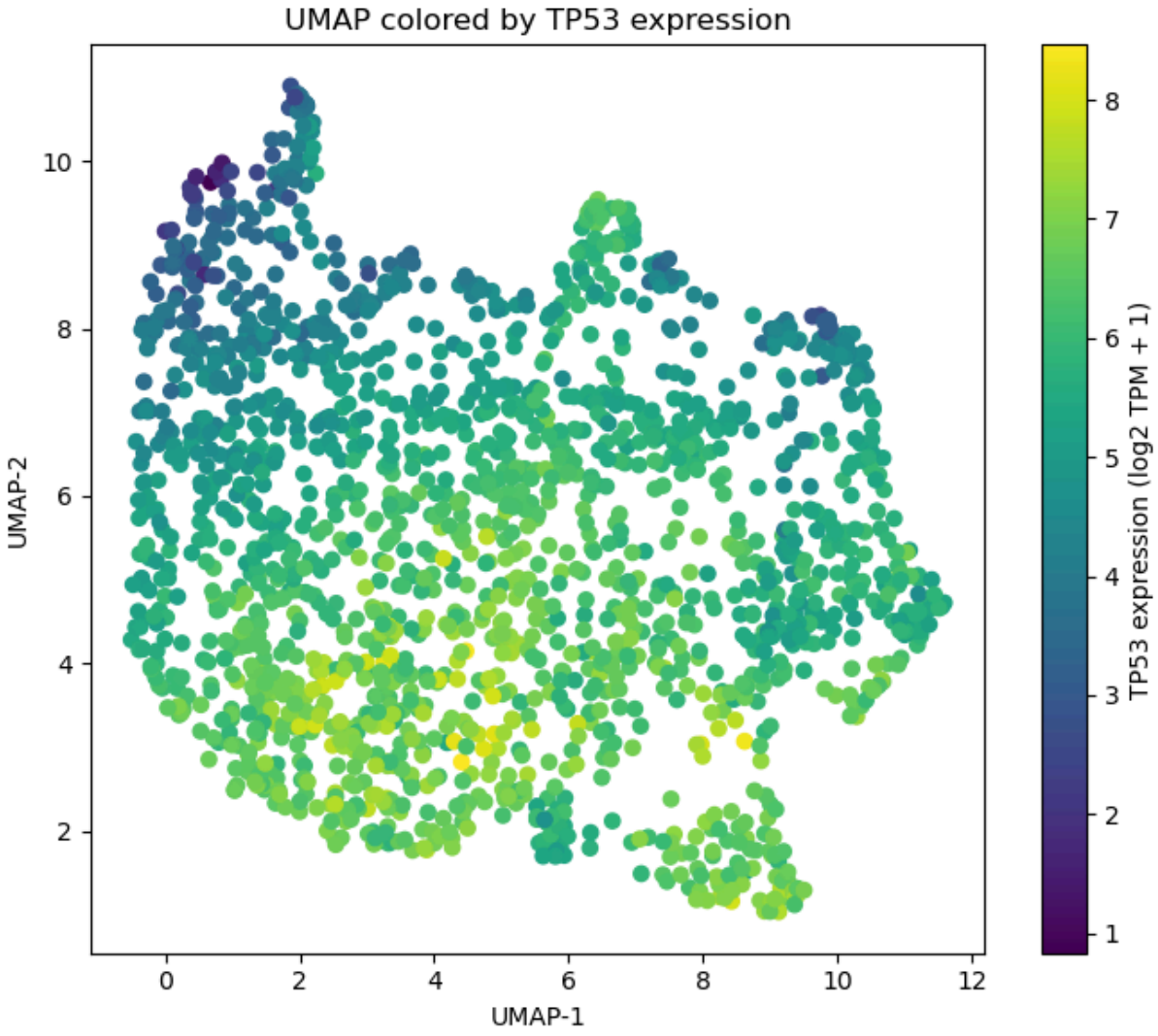
```
import pandas as pd
import matplotlib.pyplot as plt
import umap
import numpy as np

# --- Load data (note that you might have to rename file name) ---
expr = pd.read_csv("/Users/mikedornic/Desktop/bruh/UVA/BME/GSE62944_subsampl
meta = pd.read_csv("/Users/mikedornic/Desktop/bruh/UVA/BME/GSE62944_metadata

# --- Transpose (samples as rows, genes as columns) ---
expr = expr.T
expr.columns = expr.columns.astype(str).str.split('|').str[0].str.upper()

# --- Genes of interest ---
genes = ["TP53", "BCL2", "BAX", "BAK1", "CASP8"]
present = [g for g in genes if g in expr.columns]
print("Genes found:", present)

if len(present) < 2:
```

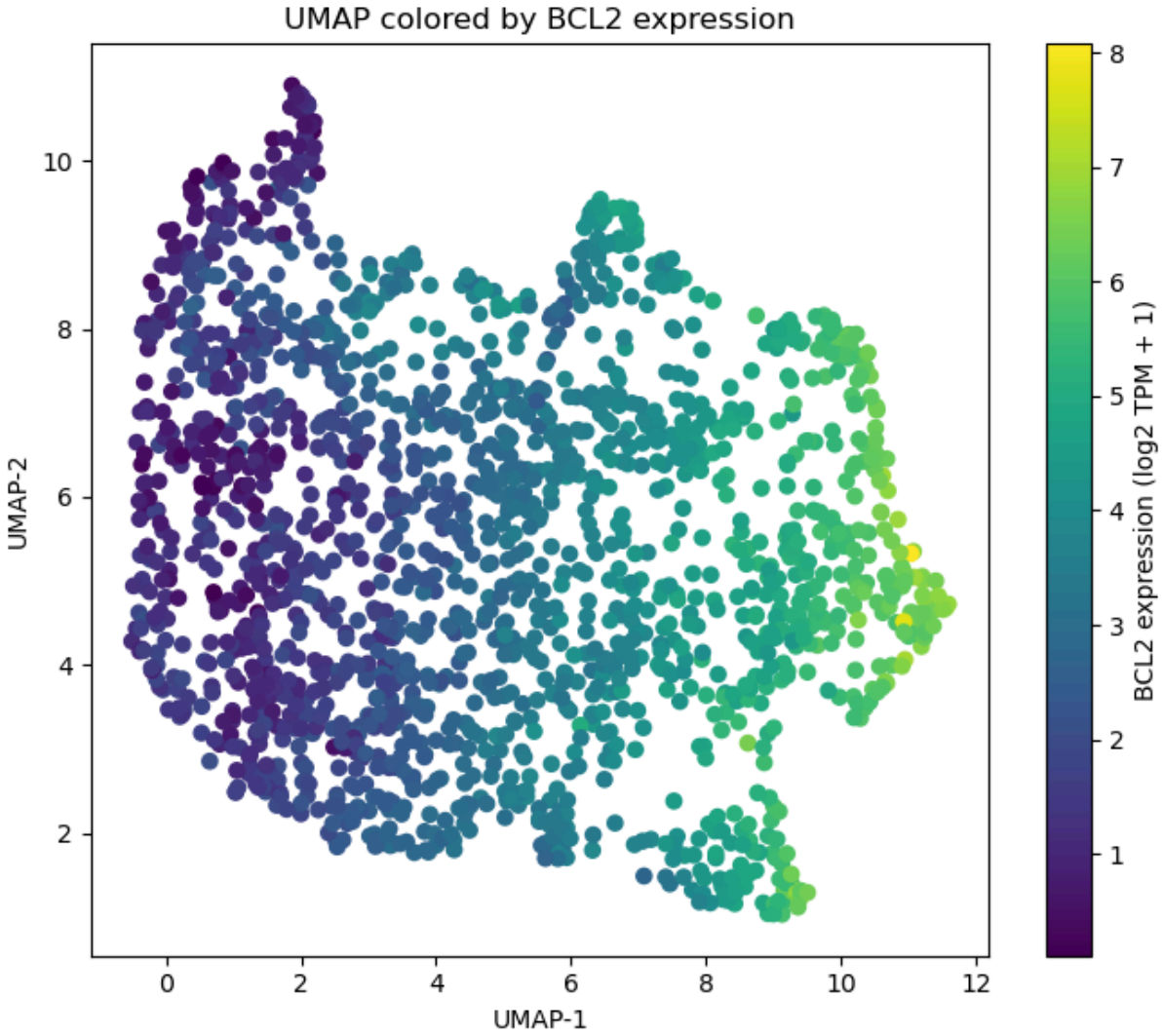
```
    raise ValueError("Need at least two genes for UMAP.")

# --- Align sample IDs ---
common = expr.index.intersection(meta.index)
expr, meta = expr.loc[common], meta.loc[common]

# --- UMAP embedding (using all apoptosis genes together) ---
X = expr[present]
reducer = umap.UMAP(n_neighbors=15, min_dist=0.3, random_state=42)
embedding = reducer.fit_transform(X)

# --- Make a color map for each gene ---
for gene in present:
    plt.figure(figsize=(7,6))
    plt.scatter(embedding[:,0], embedding[:,1],
    c=X[gene], cmap="viridis", s=35)
    plt.colorbar(label=f"{gene} expression (log2 TPM + 1)")
    plt.xlabel("UMAP-1")
    plt.ylabel("UMAP-2")
    plt.title(f"UMAP colored by {gene} expression")
    plt.tight_layout()
    plt.show()

# TEST SPLIT BELOW (TESTING HOW "ACCURATE")

from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score, calinski_harabasz_score, davie
import matplotlib.pyplot as plt

# --- Evaluate clustering quality for different k ---
def eval_kmeans_scores(Z, ks=range(2, 9), random_state=42):
    results = []
    for k in ks:
        km = KMeans(n_clusters=k, n_init='auto', random_state=random_state)
        labels = km.fit_predict(Z)

        sil = silhouette_score(Z, labels)
        ch  = calinski_harabasz_score(Z, labels)
        db  = davies_bouldin_score(Z, labels)

        results.append((k, sil, ch, db))

    # Print results
    print("k | Silhouette (↑) | Calinski-Harabasz (↑) | Davies-Bouldin (↓)")
    for k, sil, ch, db in results:
        print(f"{k:>1} | {sil:>8.3f}       | {ch:>10.1f}              | {db:

    # Pick the k with the best silhouette score
    best_k = max(results, key=lambda r: r[1])[0]
    return best_k, results

# --- Run evaluation ---
best_k, results = eval_kmeans_scores(embedding, ks=range(2, 9))

# --- Plot silhouette vs. k ---
plt.figure(figsize=(6, 4))
```

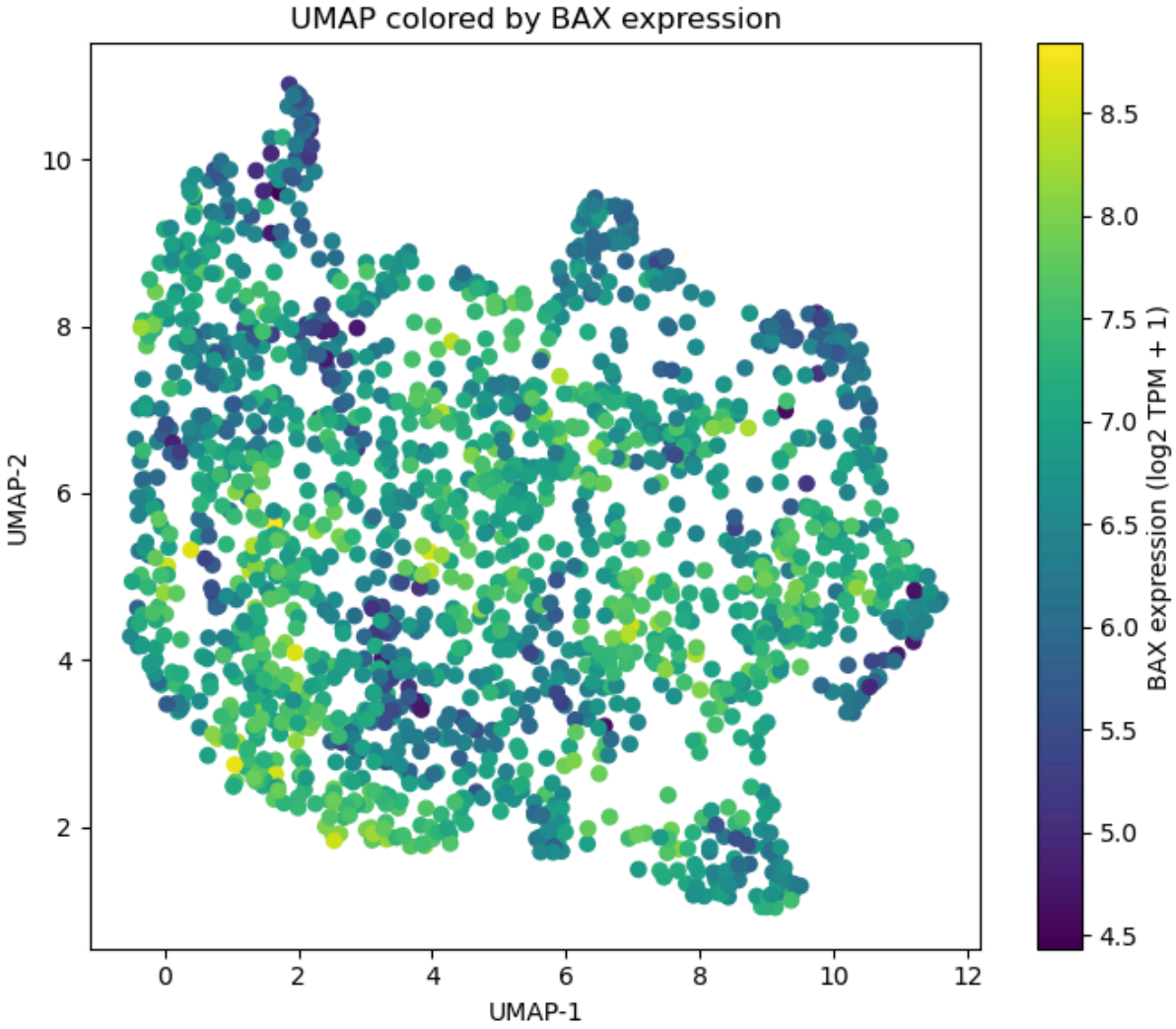
```
plt.plot([k for k, _, _, _ in results],
         [sil for _, sil, _, _ in results],
         marker='o')
plt.xlabel("Number of clusters (k)")
plt.ylabel("Silhouette score")
plt.title("Silhouette vs k (UMAP space)")
plt.tight_layout()
plt.show()

print(f"\nBest k based on silhouette score = {best_k}")
```

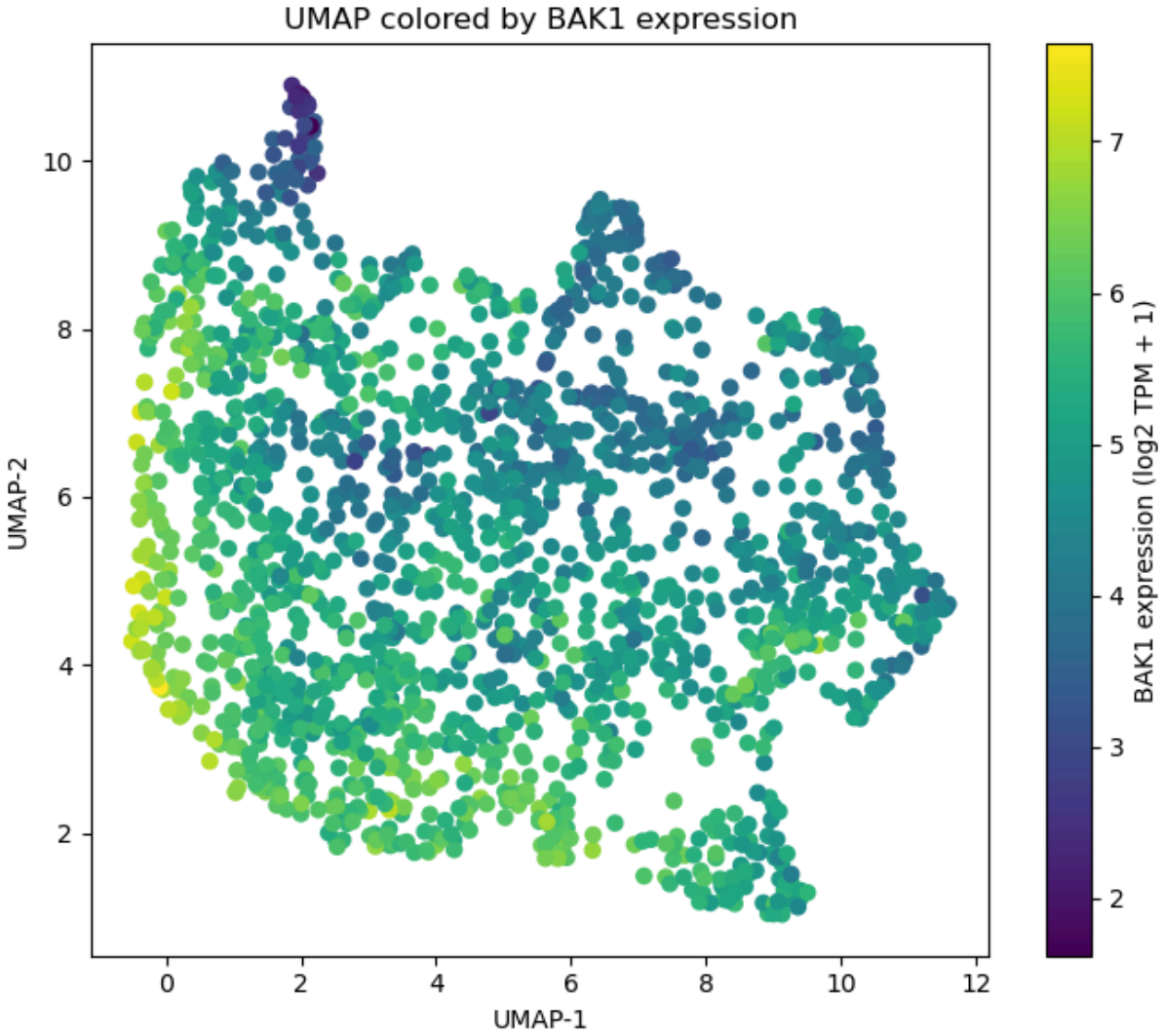
```
Genes found: ['TP53', 'BCL2', 'BAX', 'BAK1', 'CASP8']
```

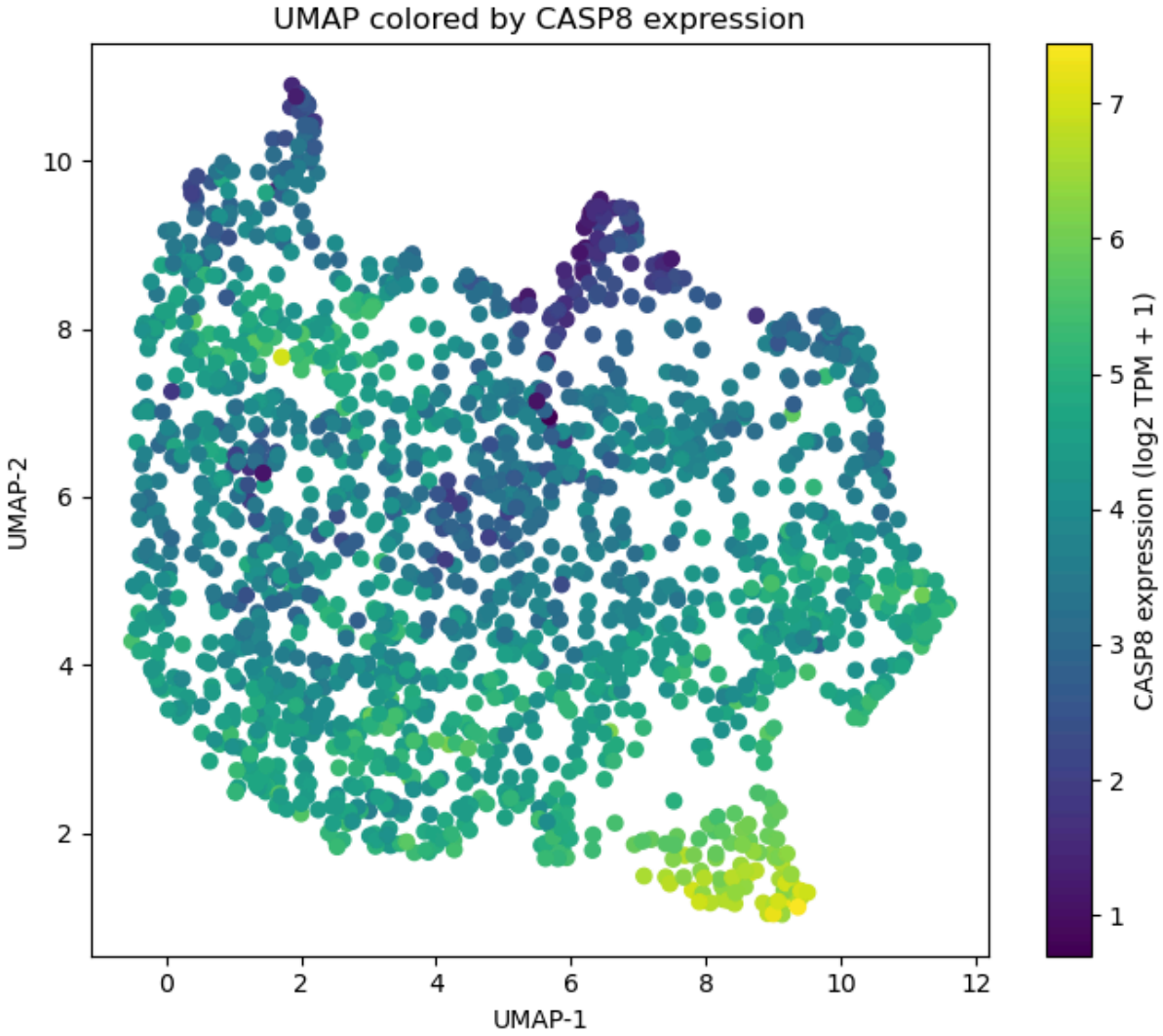
```
/opt/anaconda3/lib/python3.13/site-packages/umap/umap_.py:1952: UserWarning:
n_jobs value 1 overridden to 1 by setting random_state. Use no seed for para
llelism.
  warn(
```



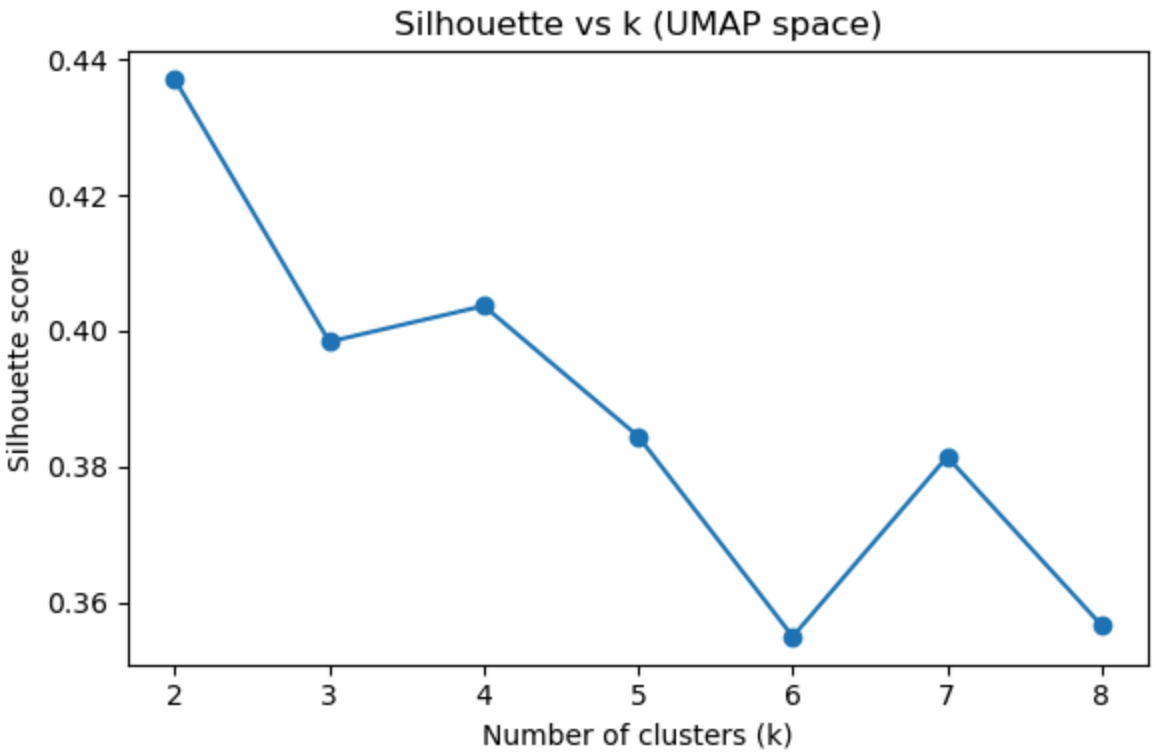

UMAP colored by BCL2 expression
UMAP-2
UMAP-1
BCL2 expression (log2 TPM + 1)
0
2
4
6
8
10
12
1
2
3
4
5
6
7
8


UMAP colored by BAX expression
UMAP-1
UMAP-2
BAX expression (log2 TPM + 1)
0
2
4
6
8
10
12
2
4
6
8
10
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
8.5


UMAP colored by BAK1 expression
UMAP-2
UMAP-1
BAK1 expression (log2 TPM + 1)
0
2
4
6
8
10
12
2
3
4
5
6
7



k	Silhouette (↑)	Calinski–Harabasz (↑)	Davies–Bouldin (↓)
2	0.437	1920.5	0.899
3	0.398	1753.8	0.870
4	0.404	1856.8	0.856
5	0.384	1867.3	0.839
6	0.355	1737.2	0.890
7	0.381	1969.5	0.829
8	0.357	1928.5	0.865




```
Best k based on silhouette score = 2
```

Data Analyis:

Methods

We are using UMAP to look at the various genes within the larger dataset. UMAP is a type of unsupervised Machine Learning model that uses dimensionality reduction to plot clusters of points on a map that are related to each other.

What is this method optimizing? How does the model decide it is "good enough"?

Our UMAP model doesn't 'optimize' but instead creates a low-dimensional representation of the data that best preserves how samples relate to one another in the original high-dimensional gene-expression space. It does this by minimizing the difference between pairwise relationships in the original data and in the 2-D embedding. Visual comparisons to other UMAPs with similar conditions should be done to ensure that our UMAP model is 'good enough.'

**

Analysis

Our code generates 5 graphs using UMAP that plots the data samples onto an arbitrary 2-D plane (hence the labels "UMAP-1" and "UMAP-2" not representing any indepedent-dependent relationship). While the clusters of data are identical for each graph, each graph is color coded based on the amount of a specific gene expression (with yellow

highlighting "hot zones"). By comparing the graphs side-by-side we can see if certain gene clusters are related to each other in terms of expression.

Our code achieves this using the following steps:

1. Load data
2. Transpose the expression matrix into an m x n matrix where m = number of patient datapoints, n = number of genes
3. Find genes of interest (thus reducing the matrix to m x 5)
4. Align sample IDs between expression and metadata datasets
5. UMAP embedding
6. Color datapoints based on each specific gene prevalence
7. Output graphs (5)

(Describe how you analyzed the data. This is where you should intersperse your Python code so that anyone reading this can run your code to perform the analysis that you did, generate your figures, etc.)

Verify and validate your analysis:

Pick a SPECIFIC method to determine how well your model is performing and describe how it works here.

After implementing our unsupervised UMAP model, we added three types of metrics to determine if our model was "accurate" and provided valid insights. In the code below, we added a sub-block that introduces a silhouette score, Calinski-Harabasz (CH) score, and Davies-Bouldin (DB) score for each k amount of clusters.

- The Silhouette score calculates the average distance from any point i to any point within its own cluster, and subtracts it from the average distance from i to any point NOT within its cluster. This value is divided by whichever maximum value is larger.
- The Calinski–Harabasz Index measures the ratio of between-cluster variance to within-cluster variance, similar to an ANOVA-style statistic that calculates clustering quality.
- The Davies-Bouldin score takes the average of pairwise "worst-case" similar clusters, to indicate how similar clusters are to each other. For each individual pairwise cluster i and j, the average distance from each point in each cluster to the centroid of the cluster is calculated. Both of these distances for cluster i and j are summed (s_i + s_j) and then divided by the distance between clusters i and j, resulting in (s_i + s_j)/d_ij. For any cluster i, this process is done for each corresponding cluster j, and the "worst-case" (highest overlap) score is kept for each cluster i. These "worst-case" values are then averaged by the amount of clusters k, resulting in our overall DB score (lower average means there is less similarity among any amount of clusters).

The code also produces a graph of these different k clusters and chooses the best amount of clusters based on the associate scores. With our initial test run, we chose somewhat arbitrary values as our parameters for the UMAP: 15 neighbors, min_distance = 0.3. When running our testing code, we got the following data:

k	Silhouette (↑)	Calinski–Harabasz (↑)	Davies–Bouldin (↓)
2	0.437	1920.5	0.899
3	0.398	1753.8	0.870
4	0.404	1856.8	0.856
5	0.384	1867.3	0.839
6	0.355	1737.2	0.890
7	0.381	1969.5	0.829
8	0.357	1928.5	0.865

Best k based on silhouette score = 2

Based on this data, the best prediction for amount of clusters in our initial UMAP model is 2, suggeseting that our projection doesn't strongly separate the clusters into identifiable or insightful 'subgroups'. We then used this data to change our UMAP parameters to the following (see output below):

Number of neighbors: 10, Min_distance = 0.15

The tweak to the number of neighbors was to emphasize local structure among subgroups, while a smaller minimum distance allowed for tighter clusters among the subgroups. With these changes we obtained the following results using the same scores:

k	Silhouette (↑)	Calinski–Harabasz (↑)	Davies–Bouldin (↓)
2	0.445	2062.2	0.875
3	0.399	1781.7	0.855
4	0.420	2006.8	0.815
5	0.443	2264.5	0.759
6	0.419	2177.2	0.741
7	0.389	2135.7	0.844
8	0.390	2065.9	0.842

Best k based on silhouette score = 2

While our code still chose two clusters as the best option, k=5 had a near identical Silhouette score (0.45% change) while having a better (13.26% lower) DB score, indicating tighter and more separated clusters, as well as a higher CB score, indicating

better separation. Therefore, with these changes we were not only able to produce more clusters than before, but tighter and more inter-related clusters, allowing us to obtain greater insights into our data and the gene expression.

Updated model with testing score code

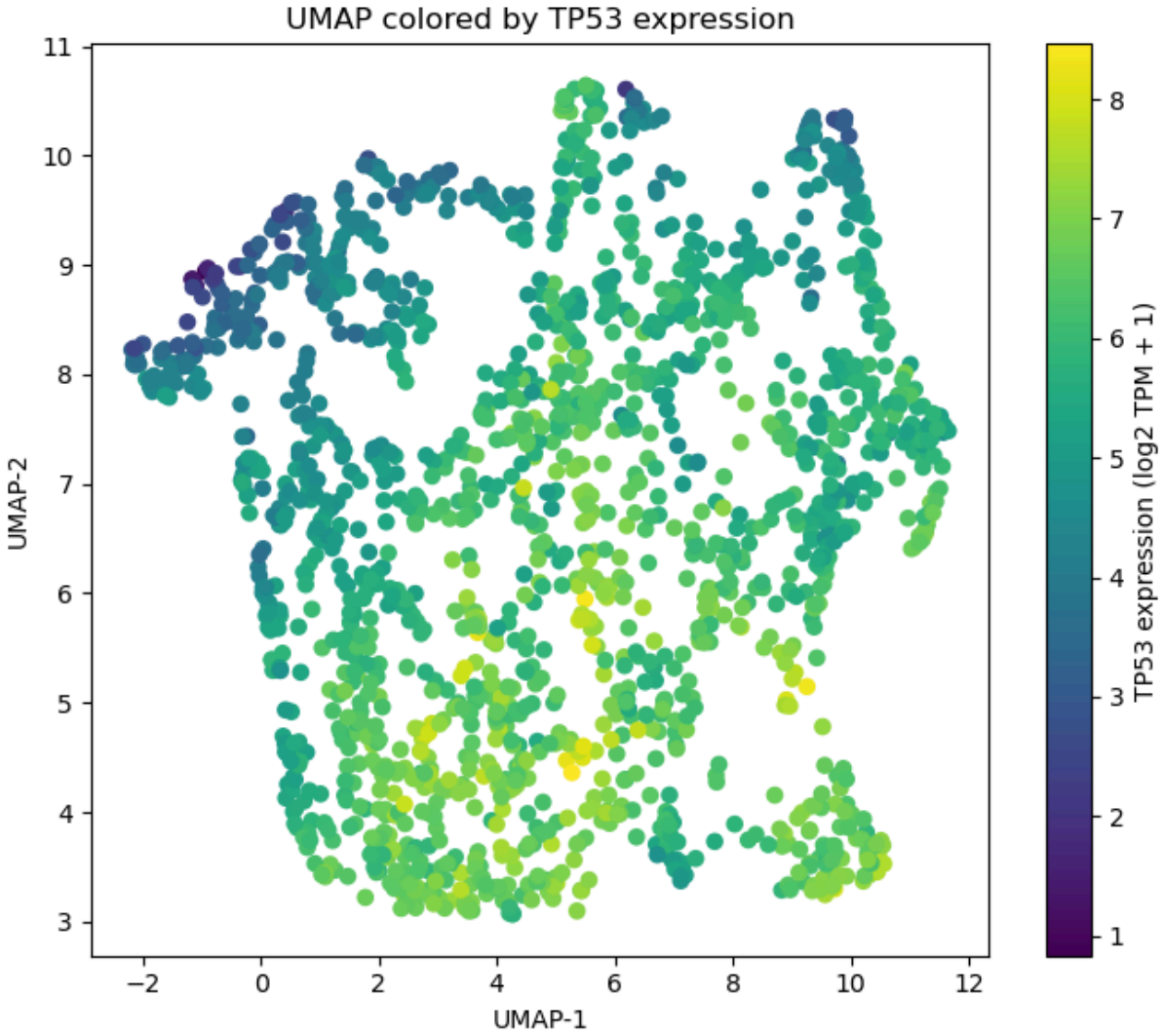
In [3]:

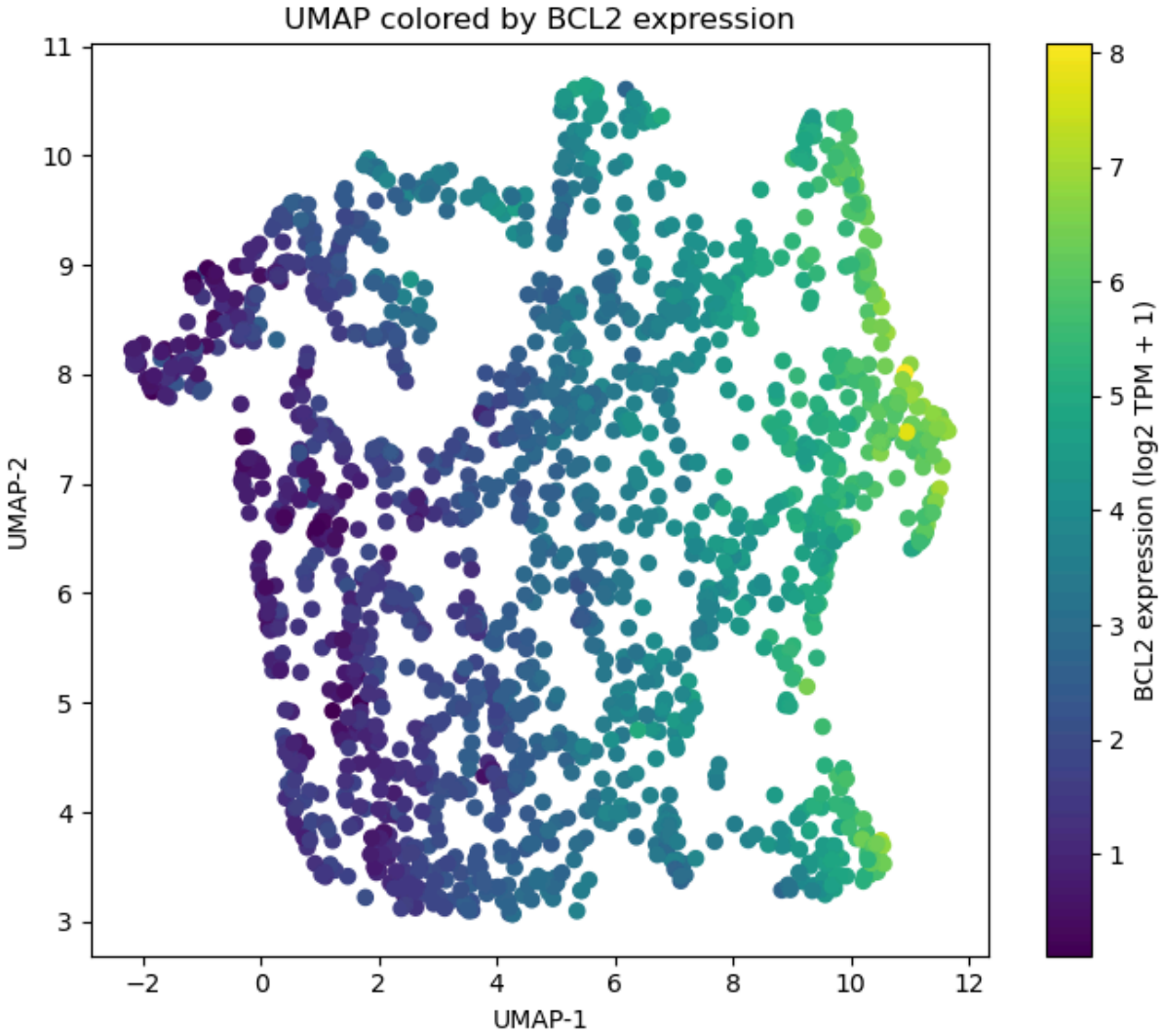
```
import pandas as pd
import matplotlib.pyplot as plt
import umap
import numpy as np

# --- Load data (note that you might have to rename file name) ---
expr = pd.read_csv("/Users/mikedornic/Desktop/bruh/UVA/BME/GSE62944_subsampl
meta = pd.read_csv("/Users/mikedornic/Desktop/bruh/UVA/BME/GSE62944_metadata

# --- Transpose (samples as rows, genes as columns) ---
expr = expr.T
expr.columns = expr.columns.astype(str).str.split('|').str[0].str.upper()

# --- Genes of interest ---
genes = ["TP53", "BCL2", "BAX", "BAK1", "CASP8"]
present = [g for g in genes if g in expr.columns]
print("Genes found:", present)

if len(present) < 2:
    raise ValueError("Need at least two genes for UMAP.")

# --- Align sample IDs ---
common = expr.index.intersection(meta.index)
expr, meta = expr.loc[common], meta.loc[common]

# --- UMAP embedding (using all apoptosis genes together) ---
X = expr[present]
reducer = umap.UMAP(n_neighbors=10, min_dist=0.15, random_state=42)
embedding = reducer.fit_transform(X)

# --- Make a color map for each gene ---
for gene in present:
    plt.figure(figsize=(7,6))
    plt.scatter(embedding[:,0], embedding[:,1],
    c=X[gene], cmap="viridis", s=35)
    plt.colorbar(label=f"{gene} expression (log2 TPM + 1)")
    plt.xlabel("UMAP-1")
    plt.ylabel("UMAP-2")
    plt.title(f"UMAP colored by {gene} expression")
    plt.tight_layout()
    plt.show()

# TEST SPLIT BELOW (TESTING HOW "ACCURATE")

from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score, calinski_harabasz_score, davie
import matplotlib.pyplot as plt
```

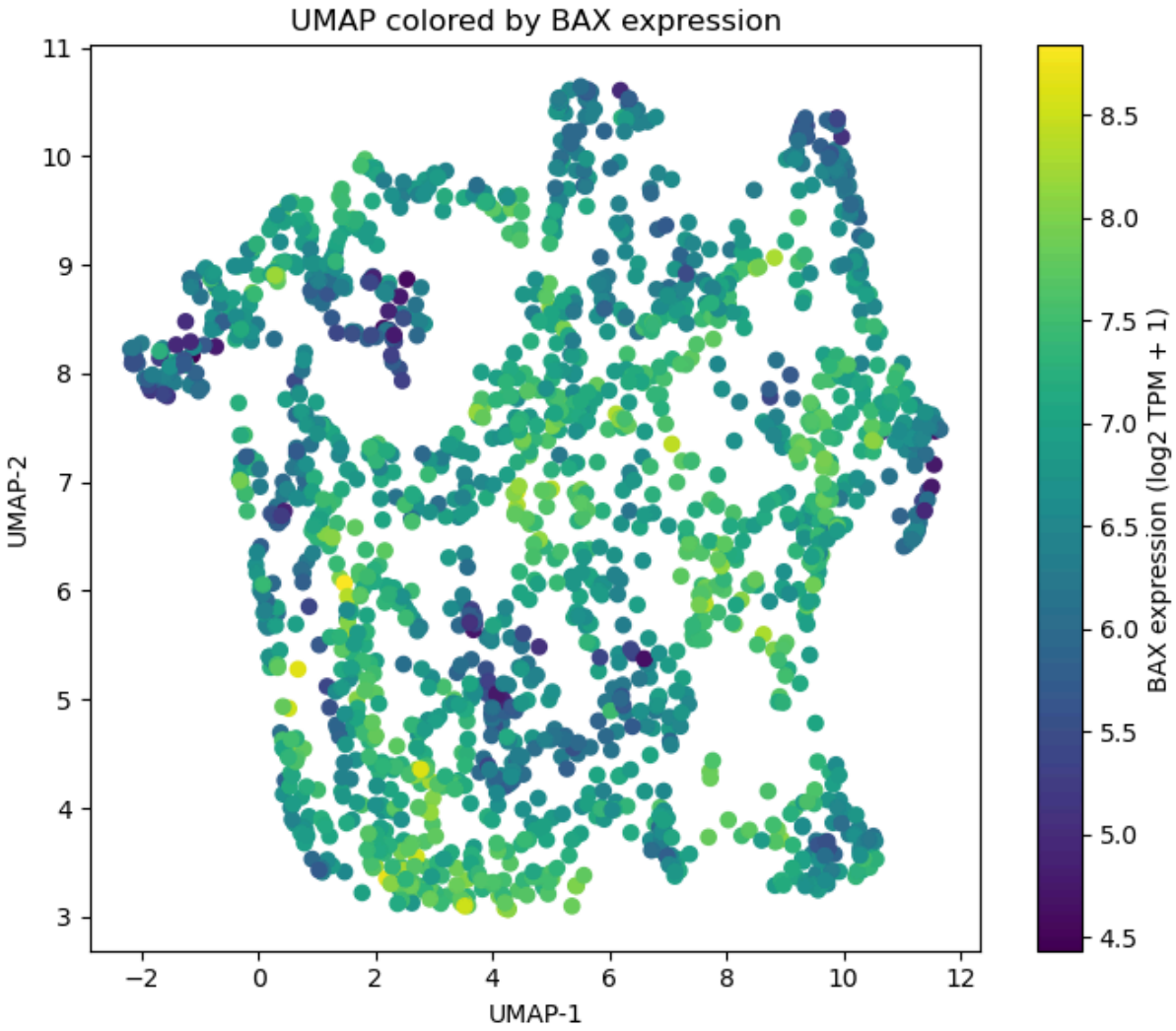
```python

# --- Evaluate clustering quality for different k ---
def eval_kmeans_scores(Z, ks=range(2, 9), random_state=42):
    results = []
    for k in ks:
        km = KMeans(n_clusters=k, n_init='auto', random_state=random_state)
        labels = km.fit_predict(Z)

        sil = silhouette_score(Z, labels)
        ch  = calinski_harabasz_score(Z, labels)
        db  = davies_bouldin_score(Z, labels)

        results.append((k, sil, ch, db))

    # Print results
    print("k | Silhouette (↑) | Calinski–Harabasz (↑) | Davies–Bouldin (↓)")
    for k, sil, ch, db in results:
        print(f"{k:>1} | {sil:>8.3f}       | {ch:>10.1f}              | {db:

    # Pick the k with the best silhouette score
    best_k = max(results, key=lambda r: r[1])[0]
    return best_k, results

# --- Run evaluation ---
best_k, results = eval_kmeans_scores(embedding, ks=range(2, 9))

# --- Plot silhouette vs. k ---
plt.figure(figsize=(6, 4))
plt.plot([k for k, _, _, _ in results],
         [sil for _, sil, _, _ in results],
         marker='o')
plt.xlabel("Number of clusters (k)")
plt.ylabel("Silhouette score")
plt.title("Silhouette vs k (UMAP space)")
plt.tight_layout()
plt.show()

print(f"\nBest k based on silhouette score = {best_k}")
```

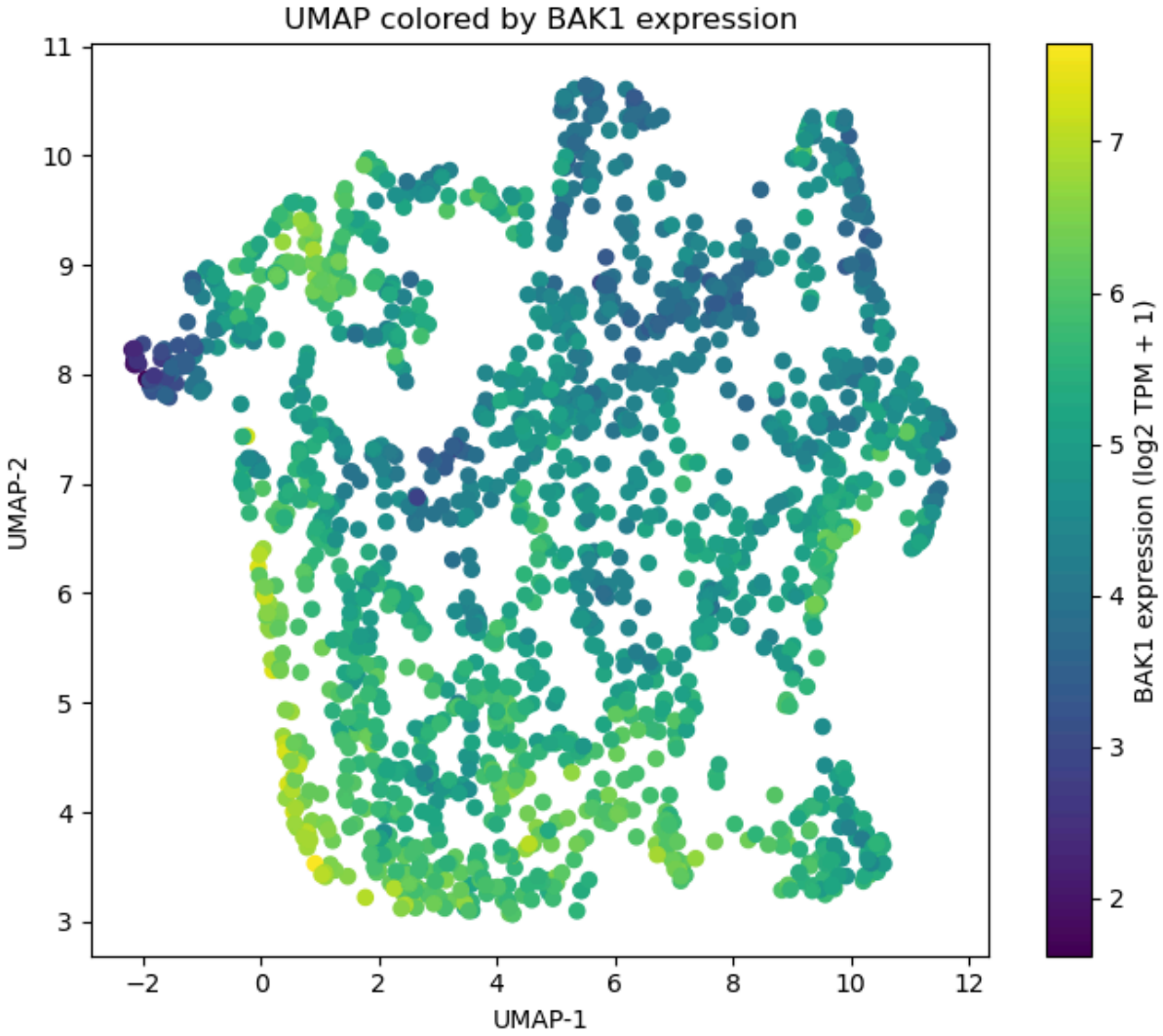
```
Genes found: ['TP53', 'BCL2', 'BAX', 'BAK1', 'CASP8']
```

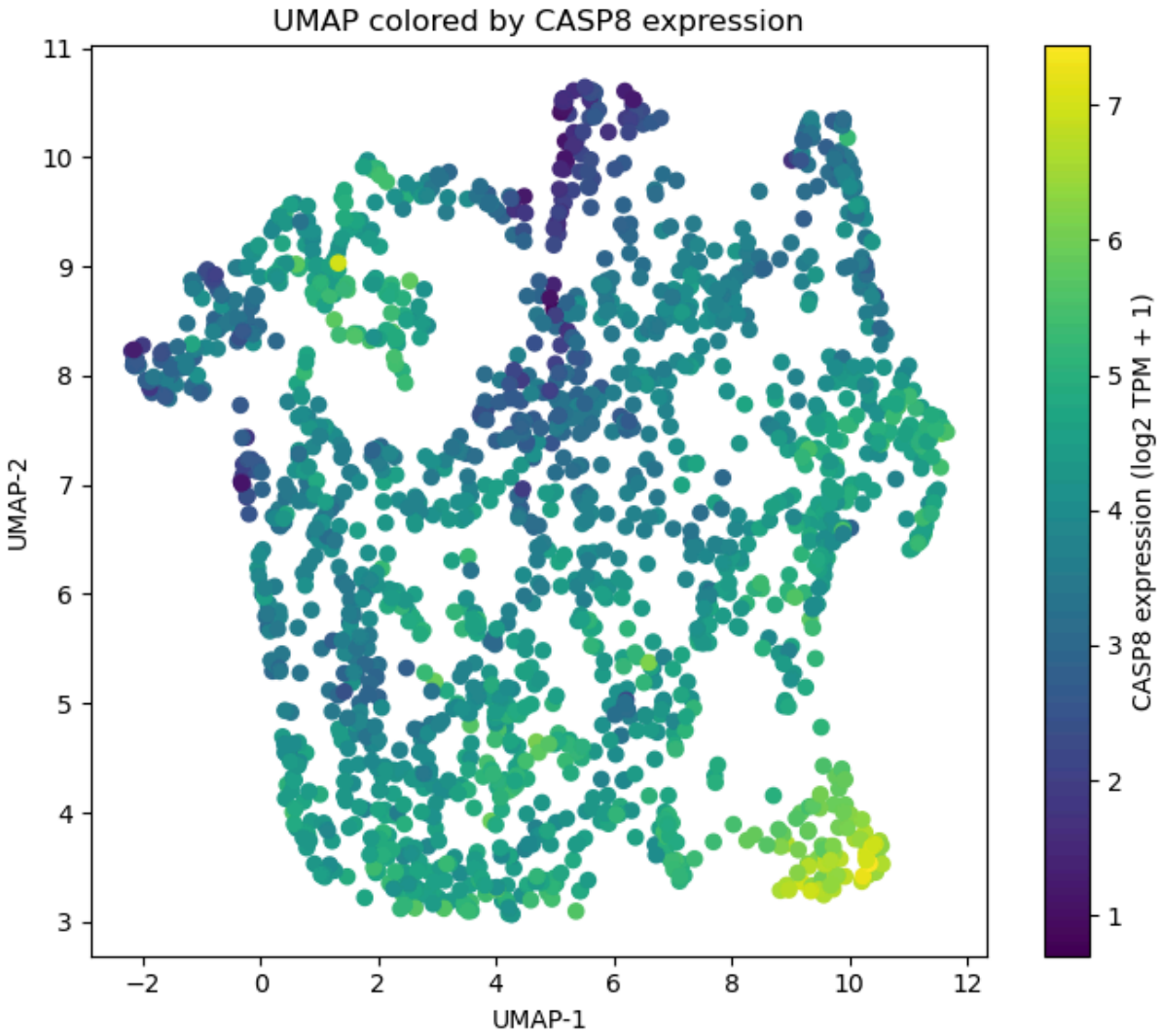
```
/opt/anaconda3/lib/python3.13/site-packages/umap/umap_.py:1952: UserWarning:
n_jobs value 1 overridden to 1 by setting random_state. Use no seed for para
llelism.
  warn(
```


UMAP colored by TP53 expression
UMAP-2
UMAP-1
TP53 expression (log2 TPM + 1)
11
10
9
8
7
6
5
4
3
−2
0
2
4
6
8
10
12
1
2
3
4
5
6
7
8

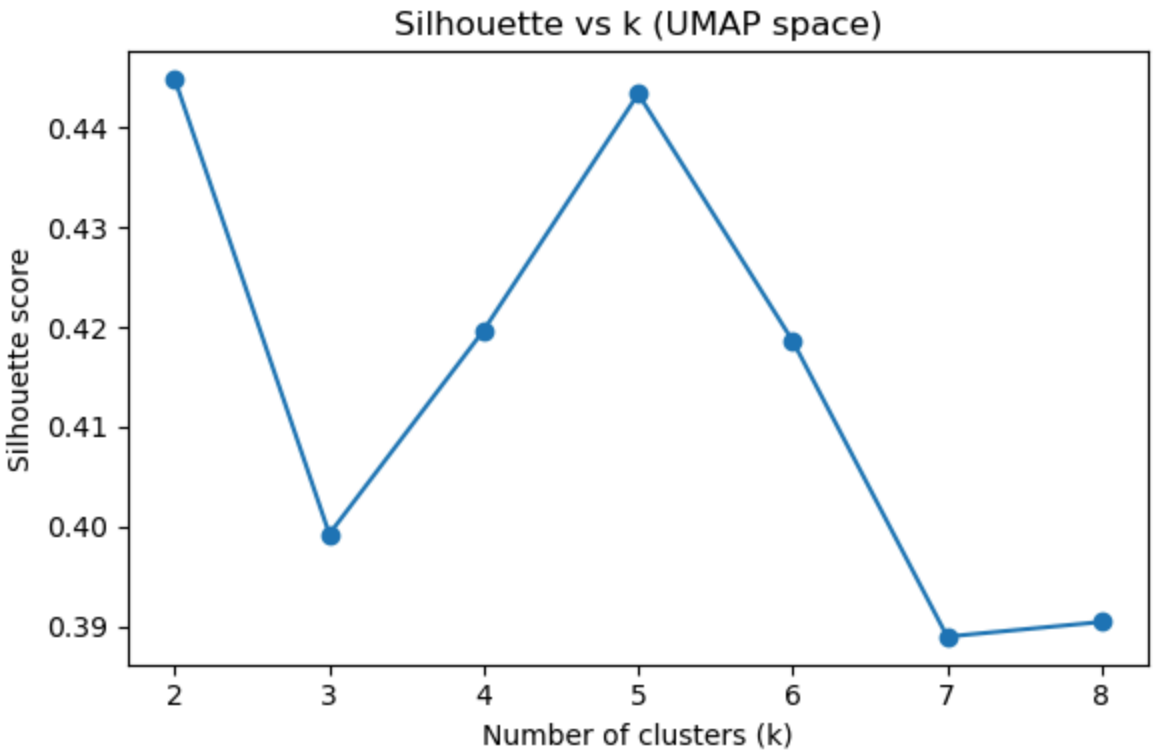

UMAP colored by BCL2 expression
UMAP-2
UMAP-1
BCL2 expression (log2 TPM + 1)
11
10
9
8
7
6
5
4
3
−2
0
2
4
6
8
10
12
8
7
6
5
4
3
2
1


UMAP colored by BAX expression
UMAP-2
UMAP-1
BAX expression (log2 TPM + 1)
11
10
9
8
7
6
5
4
3
−2
0
2
4
6
8
10
12
8.5
8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5


UMAP colored by BAK1 expression
UMAP-2
UMAP-1
BAK1 expression (log2 TPM + 1)
11
10
9
8
7
6
5
4
3
−2
0
2
4
6
8
10
12
7
6
5
4
3
2



k	Silhouette (↑)	Calinski–Harabasz (↑)	Davies–Bouldin (↓)
2	0.445	2062.2	0.875
3	0.399	1781.7	0.855
4	0.420	2006.8	0.815
5	0.443	2264.5	0.759
6	0.419	2177.2	0.741
7	0.389	2135.7	0.844
8	0.390	2065.9	0.842




```
Best k based on silhouette score = 2
```

Conclusions and Ethical Implications:

Based on our graphs, data samples with a high amount of the CASP8 gene form their own mini cluster in the bottom right corner, which is the only separate cluster from the larger blob. Therefore, future iterations may require us to redo the UMAP or consider other unsupervised learning models such as PCA to get better visualizations with more separated clusters. In terms of color separation, the 'clearest' graph visually was the BCL2 expression graph, where you can see a disjoint-looking set of colors from purple (far left) to yellow (far right). The UMAP graph colored for BAK1 also visually appears to be the 'inverse' of BCL2, as the yellow zones are on opposite sides of the distribution, suggesting some type of inverse relationship. This suggests, but does not prove that our model was able to 'correctly' map the data onto two-dimensions based on related genes.

It is important to note that while our model 'found' interesting relationships between CASP8, BCL2 for the datapoints, because it was unsupervised, these findings may or may not have any valid implications. In other words, the distribution of the dataset under our model may be biased by other factors not accounted for in the model. For example, our model only looked at 5 genes instead of the entire distribution of genes for breast cancer. As a result, future work should be done on the entire breast cancer dataset to see if BCL2 still "drives" the manifold (i.e. are the UMAP clusters still distrinbuted in a discrete color continuum).

Limitations and Future Work:

After implementing our testing code block using the different scores, we were able to review our initial parameters for UMAP and improve them to produce more separated and tighter clusters, revealing greater structure among the genes. To optimize our model, we could either test additional parameter combinations (less neighbors, smaller min distance etc.) or create a local variable that holds the "average mean score" for each parameter combination (initially set to 0), then iterate over each parameter combination by some gradual change k in a for loop (we can make k as large or small as possible depending on desired accuracy), replacing the variable with the "best score" using if else statements within the loop to find the most optimal paramater combination for our model. Using this "optimized" combination, we can then test our UMAP model on different datasets (beyond the GSE62944 dataset) to see how it compares to already existing UMAP or PCA unsupervised models.

NOTES FROM YOUR TEAM:

This is where our team is taking notes and recording activity.

QUESTIONS FOR YOUR TA:

What might the findings of the BCL2 graph suggest about the individual datasamples? Could this mean if the datasamples did not contain BCL2, then they were not related to each other via another gene?