

Enhanced Early Cancer Detection via Topological Data Analysis of Blood Biomarkers: A Non-Invasive Approach for Accurate, Multi-Cancer Screening

Sudarshan Gogoi^a, Bikash Thakuri^b, Amit Chakraborty^{c,*}

^{a,,c} Department of Mathematics, Sikkim University, Gangtok-737102, Sikkim, India

^b School of Basic Sciences, SRM University, Sikkim

Tutorial on Biomarker Selection, Cancer Detection, and Tissue Localization Process

1. Open the link: <https://github.com/Sudarshan-Gogoi/TDA-Detect-Cancer/tree/main> .

2. Click on Codes and files

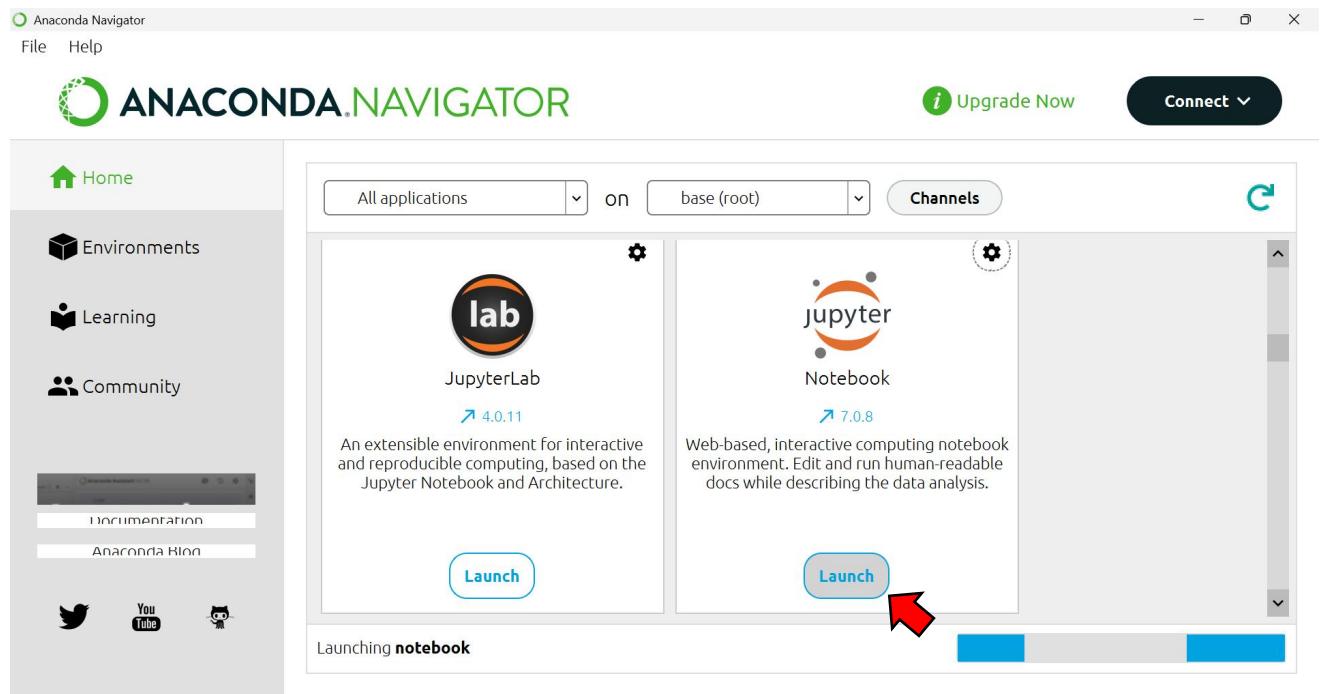
3. Download Files:

- “Optimized Biomarker Selection.ipynb”
- “TDAcancer_detect.ipynb”
- “Clinical cancer data.xlsx” (Example File)

4. Open Anaconda Navigator.

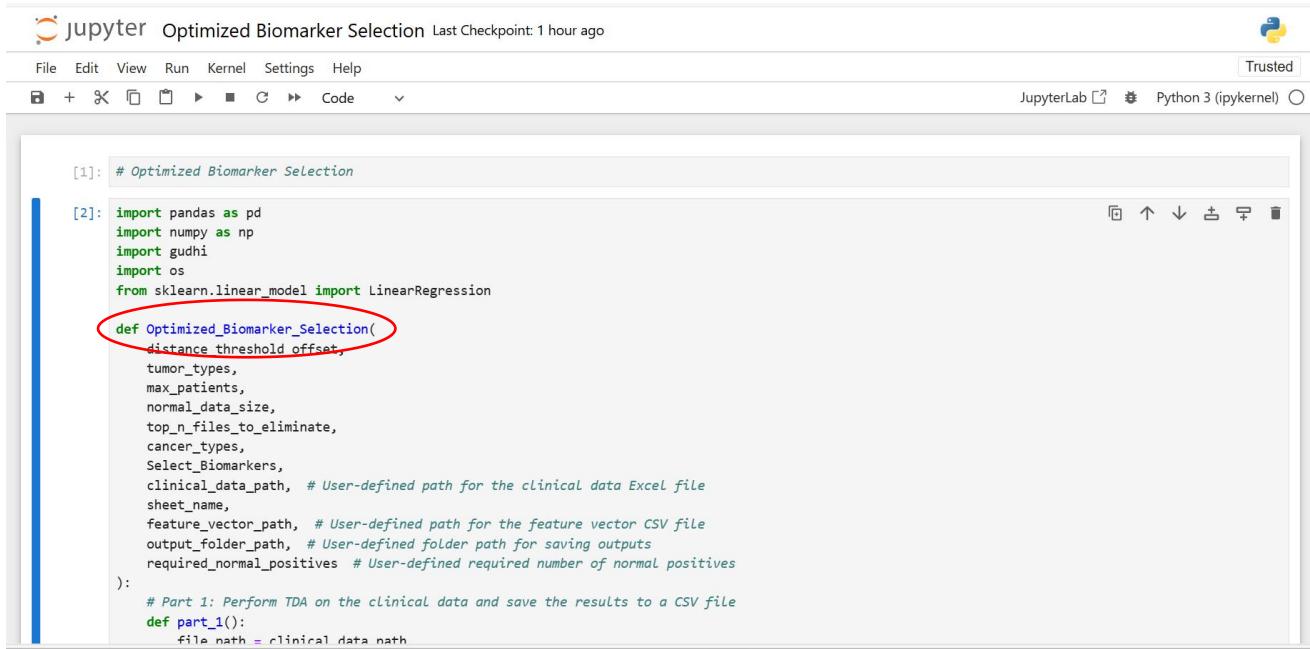
5. Launch Jupyter Notebook (Version 7.0.8):

- Click on Launch under Jupyter Notebook.



Biomarker Selection Process

6. Open Jupyter Notebook and load “Optimized Biomarker Selection.ipynb”. (If the user wants to use the default example biomarker set for analysis, then directly move to the step. 28)



The screenshot shows a Jupyter Notebook interface with the title "Optimized Biomarker Selection". The code cell [2] contains Python code for biomarker selection. A red oval highlights the function definition:

```
[1]: # Optimized Biomarker Selection
[2]: import pandas as pd
       import numpy as np
       import gudhi
       import os
       from sklearn.linear_model import LinearRegression

def Optimized_Biomarker_Selection(
    distance_threshold_offset,
    tumor_types,
    max_patients,
    normal_data_size,
    top_n_files_to_eliminate,
    cancer_types,
    Select_Biomarkers,
    clinical_data_path, # User-defined path for the clinical data Excel file
    sheet_name,
    feature_vector_path, # User-defined path for the feature vector CSV file
    output_folder_path, # User-defined folder path for saving outputs
    required_normal_positives # User-defined required number of normal positives
):
    # Part 1: Perform TDA on the clinical data and save the results to a CSV file
    def part_1():
        file_path = clinical_data_path
```

7. Enter the required inputs to the function ***Optimized_Biomarker_Selection***. The input descriptions are given below:

- **distance_threshold_offset**: Enter a value between 0 and 2.
- **tumor_types**: List the tumor types present in the dataset along with the number of patients to use as training samples for each tumor.
- **max_patients**: Specify the total number of patients in the dataset.
- **normal_data_size**: Provide the number of healthy individuals to use as training samples.
- **top_n_files_to_eliminate**: Mention the number of least significant biomarkers to eliminate from the set of biomarkers present in the dataset.
- **cancer_types**: List the cancer types present in the dataset.
- **Select_Biomarkers**: List the biomarkers present in the dataset.
- ***clinical_data_path**: Path to the file containing the Excel data file “**Clinical cancer data.xlsx**”.
- **sheet_name**: Input the patient data sheet “**Normal and Cancer**” from the provided Excel data file “**Clinical cancer data.xlsx**”.
- **feature_vector_path**: Path to store the feature vector file.
- **output_folder_path**: Path to store the intermediate result files.
- **required_normal_positives**: Minimum number of cancer false positives that can be considered to make specificity above 95%.

***Note:** For the example file “Clinical cancer data.xlsx”, the user must provide **distance_threshold_offset**, **top_n_files_to_eliminate**, **clinical_data_path**, **feature_vector_path**, and **output_folder_path**. The other inputs are set as default. For **distance_threshold_offset** and **top_n_files_to_eliminate**, the user can use values from **Table 1** (given below) as default. The inputs

are case-sensitive, so refer to the example inputs and ensure that your inputs match the given case exactly.

8. Run the function *Optimize_Biomarker_Selection*.

```

jupyter Optimized Biomarker Selection Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help Trusted
JupyterLab Python 3 (ipykernel) 2

print(f"The Significant Biomarkers: {remaining_biomarkers}")

eliminated_biomarkers = [biomarker_file_map[file_name] for file_name, _ in top_files if file_name in biomarker_file_map]
print(f"Eliminated least significant (top_n_files_to_eliminate) biomarkers: {eliminated_biomarkers}")

# Execute the parts in sequence
part_1()
Normal_threshold = part_2()
biomarker_file_map, file_biomarker_map = part_3()
part_4(biomarker_file_map, file_biomarker_map)

# Run the function with user-defined values
Optimized_Biomarker_Selection(
    distance_threshold_offset=0.7,
    tumor_types={'normal': 554, 'breast': 156, 'colorectum': 291, 'esophagus': 34, 'liver': 33, 'lung': 78, 'ovary': 40, 'pancreas': 69, 'stomach': 50},
    max_patients=1882,
    normal_data_size=554,
    top_n_files_to_eliminate=2,
    cancer_types=['Breast', 'Colorectum', 'Esophagus', 'Liver', 'Lung', 'Ovary', 'Pancreas', 'Stomach'],
    Select_Biomarkers=[ 'AFP', 'Angiopoietin-2', 'AXL', 'CA-125', 'CA 15-3', 'CA19-9', 'CD44', 'CEA', 'CYFRA 21-1', 'DKK1', 'Endoglin', 'FGF2', 'Follistatin', 'GATA3', 'ICAM-1', 'IL-6', 'Lactate dehydrogenase', 'MMP-9', 'Nanog', 'PAP', 'Prostate-specific antigen', 'SFRP1', 'Tissue polypeptide isomerase', 'VEGFR-2', 'Vimentin'],
    clinical_data_path='D:/Cancer Detection folder/Biomarker Selection/Clinical cancer data.xlsx',
    sheet_name = 'Normal and Cancer',
    feature_vector_path="D:/Cancer Detection folder/Biomarker Selection/FeatureVectorCancerDet.csv",
    output_folder_path="D:/Cancer Detection folder/Biomarker Selection",
    required_normal_positives=36 # User-defined required number of normal positives
)

```

9. Observe the Total Positive Cancers number.

```

jupyter Optimized Biomarker Selection Last Checkpoint: 20 hours ago
File Edit View Run Kernel Settings Help Trusted
JupyterLab Python 3 (ipykernel) 2

1-Dim Hole Max Life Range Average Birth (1D) Average Death (1D) \
0 0.157292 0.710014 0.754830
1 0.156110 0.709123 0.756199
2 0.154643 0.710634 0.758400
3 0.148888 0.708844 0.756571
4 0.150139 0.704365 0.753228
5 0.162510 0.708667 0.756623
6 0.161601 0.709348 0.758956
7 0.167336 0.705547 0.755778
8 0.158705 0.706479 0.756383

Regression Difference Percentage Distance %(1)
0 0.729729 0.830194
1 0.871621 1.060106
2 0.931489 1.113619
3 1.198704 1.017403
4 1.798894 1.392264
5 0.776544 1.132190
6 1.735678 1.070448
7 1.236223 1.361441
8 1.348752 1.457229

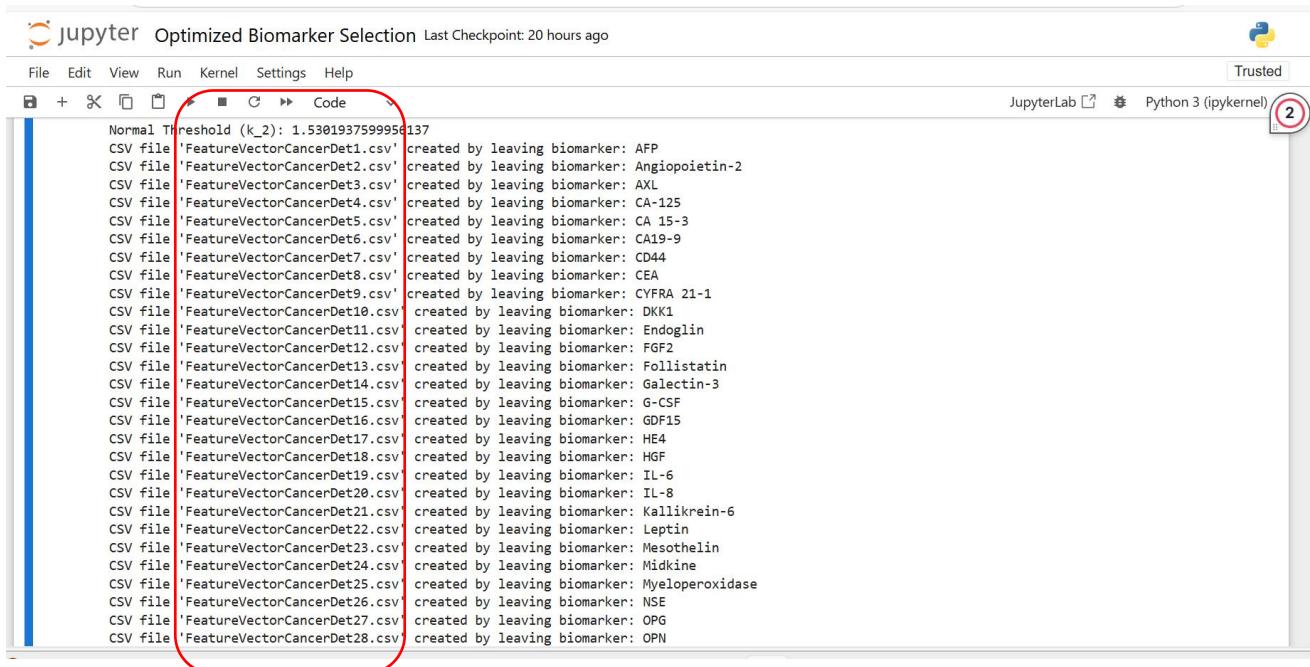
Final 'Distance %(1)' Threshold (k_2): 1.530194
Final Regression Difference Threshold (k_1): 0.6820
Total Positive Cancers: 141
Normal Threshold (k_2): 1.5301937599956137
CSV file 'FeatureVectorCancerDet1.csv' created by leaving biomarker: AFP

```

10. If the Total Positive Cancers number is not satisfactory, proceed to the next step.

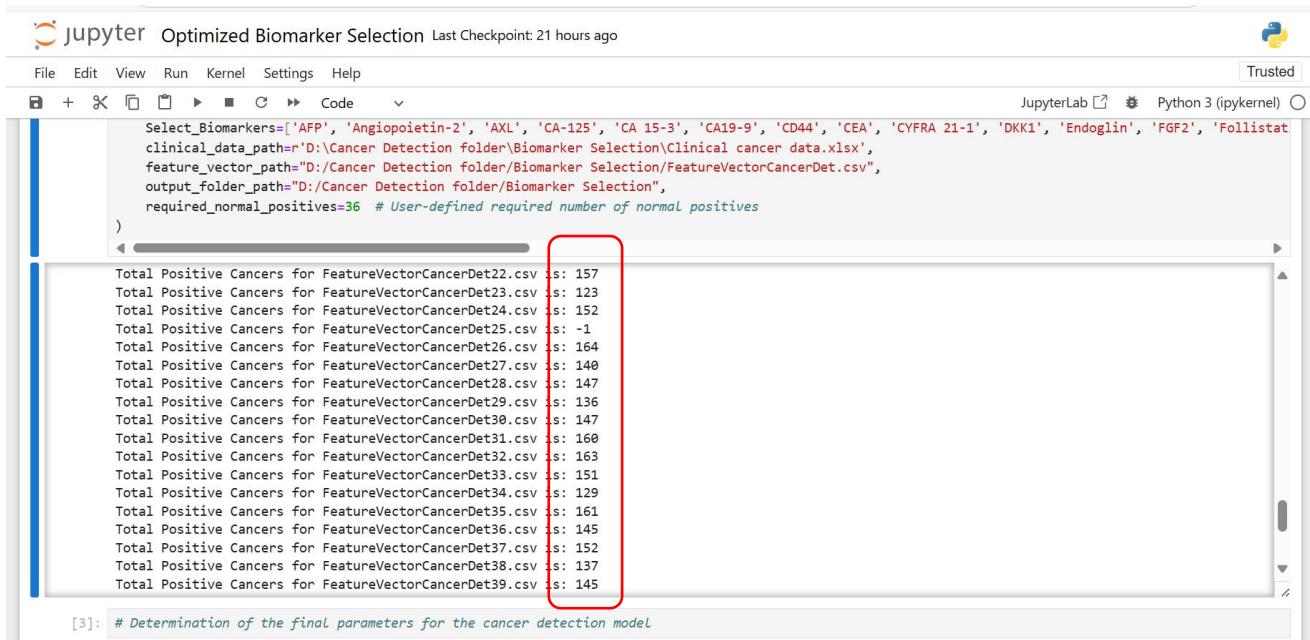
11. A number of CSV files (e.g., FeatureVectorCancerDet1, FeatureVectorCancerDet2, etc.) will be generated and saved in the given “**output_folder_path**” for required analysis.

Biomarker Selection Process



```
Normal Threshold (k_2): 1.530193759995137
CSV file 'FeatureVectorCancerDet1.csv' created by leaving biomarker: AFP
CSV file 'FeatureVectorCancerDet2.csv' created by leaving biomarker: Angiopoietin-2
CSV file 'FeatureVectorCancerDet3.csv' created by leaving biomarker: AXL
CSV file 'FeatureVectorCancerDet4.csv' created by leaving biomarker: CA-125
CSV file 'FeatureVectorCancerDet5.csv' created by leaving biomarker: CA 15-3
CSV file 'FeatureVectorCancerDet6.csv' created by leaving biomarker: CA19-9
CSV file 'FeatureVectorCancerDet7.csv' created by leaving biomarker: CD44
CSV file 'FeatureVectorCancerDet8.csv' created by leaving biomarker: CEA
CSV file 'FeatureVectorCancerDet9.csv' created by leaving biomarker: CYFRA 21-1
CSV file 'FeatureVectorCancerDet10.csv' created by leaving biomarker: DKK1
CSV file 'FeatureVectorCancerDet11.csv' created by leaving biomarker: Endoglin
CSV file 'FeatureVectorCancerDet12.csv' created by leaving biomarker: FGF2
CSV file 'FeatureVectorCancerDet13.csv' created by leaving biomarker: Follistatin
CSV file 'FeatureVectorCancerDet14.csv' created by leaving biomarker: Galectin-3
CSV file 'FeatureVectorCancerDet15.csv' created by leaving biomarker: G-CSF
CSV file 'FeatureVectorCancerDet16.csv' created by leaving biomarker: GDF15
CSV file 'FeatureVectorCancerDet17.csv' created by leaving biomarker: HE4
CSV file 'FeatureVectorCancerDet18.csv' created by leaving biomarker: HGF
CSV file 'FeatureVectorCancerDet19.csv' created by leaving biomarker: IL-6
CSV file 'FeatureVectorCancerDet20.csv' created by leaving biomarker: IL-8
CSV file 'FeatureVectorCancerDet21.csv' created by leaving biomarker: Kallikrein-6
CSV file 'FeatureVectorCancerDet22.csv' created by leaving biomarker: Leptin
CSV file 'FeatureVectorCancerDet23.csv' created by leaving biomarker: Mesothelin
CSV file 'FeatureVectorCancerDet24.csv' created by leaving biomarker: Midkine
CSV file 'FeatureVectorCancerDet25.csv' created by leaving biomarker: Myeloperoxidase
CSV file 'FeatureVectorCancerDet26.csv' created by leaving biomarker: NSE
CSV file 'FeatureVectorCancerDet27.csv' created by leaving biomarker: OPG
CSV file 'FeatureVectorCancerDet28.csv' created by leaving biomarker: OPN
```

12. The Total Positive Cancers number for each CSV file will be shown. -1 indicates that no cancer positives were obtained for the given threshold.



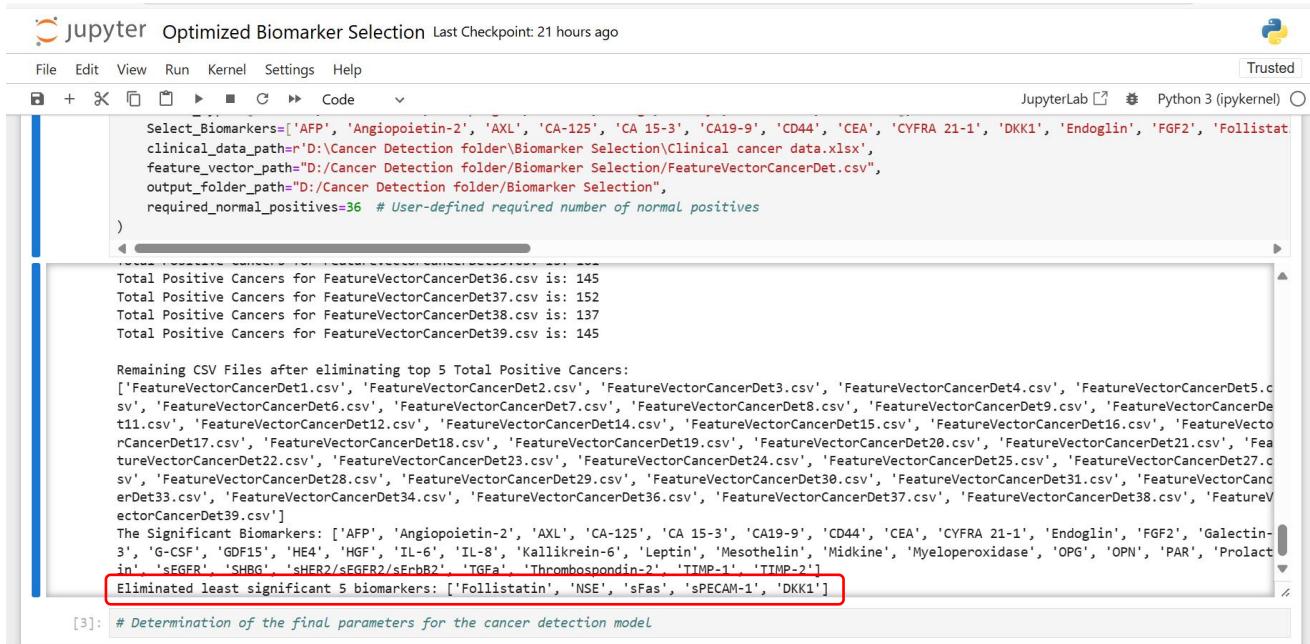
```
Select_Biomarkers=['AFP', 'Angiopoietin-2', 'AXL', 'CA-125', 'CA 15-3', 'CA19-9', 'CD44', 'CEA', 'CYFRA 21-1', 'DKK1', 'Endoglin', 'FGF2', 'Follistatin', 'G-CSF', 'GDF15', 'HE4', 'HGF', 'IL-6', 'IL-8', 'Kallikrein-6', 'Leptin', 'Mesothelin', 'Midkine', 'Myeloperoxidase', 'NSE', 'OPG', 'OPN']
clinical_data_path=r'D:\Cancer Detection folder\Biomarker Selection\Clinical cancer data.xlsx',
feature_vector_path="D:/Cancer Detection folder/Biomarker Selection/FeatureVectorCancerDet.csv",
output_folder_path="D:/Cancer Detection folder/Biomarker Selection",
required_normal_positives=36 # User-defined required number of normal positives
)

Total Positive Cancers for FeatureVectorCancerDet22.csv : 157
Total Positive Cancers for FeatureVectorCancerDet23.csv : 123
Total Positive Cancers for FeatureVectorCancerDet24.csv : 152
Total Positive Cancers for FeatureVectorCancerDet25.csv : -1
Total Positive Cancers for FeatureVectorCancerDet26.csv : 164
Total Positive Cancers for FeatureVectorCancerDet27.csv : 140
Total Positive Cancers for FeatureVectorCancerDet28.csv : 147
Total Positive Cancers for FeatureVectorCancerDet29.csv : 136
Total Positive Cancers for FeatureVectorCancerDet30.csv : 147
Total Positive Cancers for FeatureVectorCancerDet31.csv : 160
Total Positive Cancers for FeatureVectorCancerDet32.csv : 163
Total Positive Cancers for FeatureVectorCancerDet33.csv : 151
Total Positive Cancers for FeatureVectorCancerDet34.csv : 129
Total Positive Cancers for FeatureVectorCancerDet35.csv : 161
Total Positive Cancers for FeatureVectorCancerDet36.csv : 145
Total Positive Cancers for FeatureVectorCancerDet37.csv : 152
Total Positive Cancers for FeatureVectorCancerDet38.csv : 137
Total Positive Cancers for FeatureVectorCancerDet39.csv : 145
```

13. The least significant biomarkers will be eliminated based on the number of **top_n_files_to_eliminate** selected by the user.

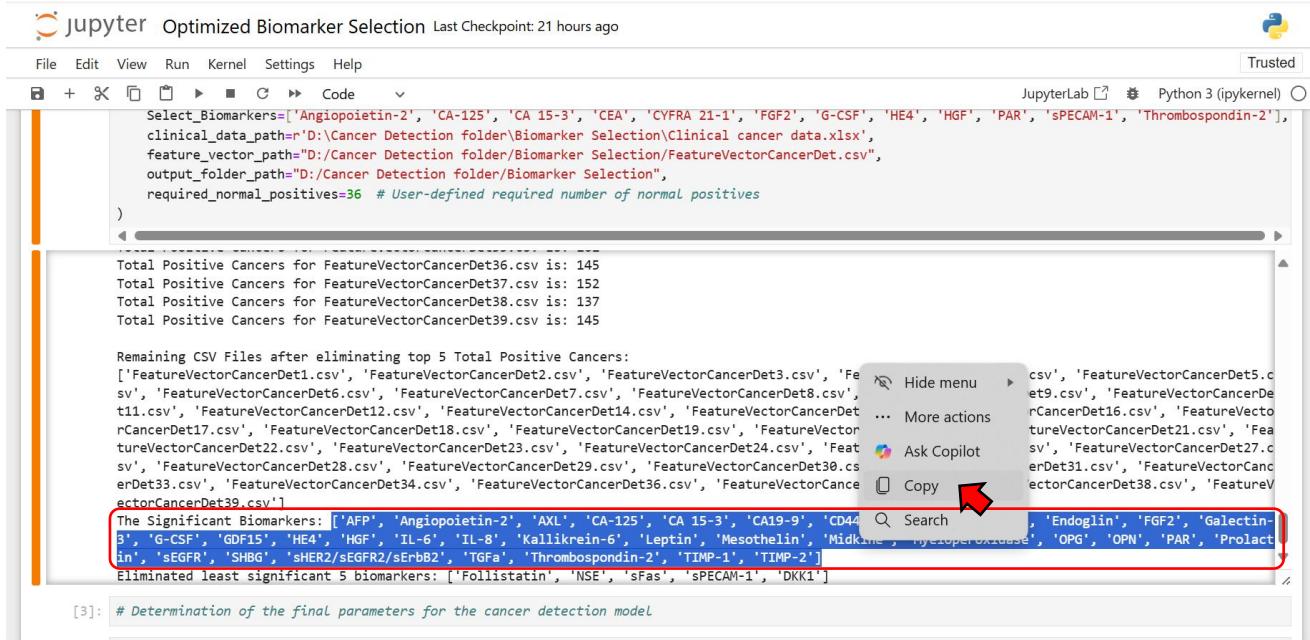
14. **top_n_files_to_eliminate** will include those biomarkers related to the CSV files (e.g., FeatureVectorCancerDet1, FeatureVectorCancerDet2, etc.) with the top n Total Positive Cancers numbers.

Biomarker Selection Process



```
Select_Biomarkers=['AFP', 'Angiopoietin-2', 'AXL', 'CA-125', 'CA 15-3', 'CA19-9', 'CD44', 'CEA', 'CYFRA 21-1', 'DKK1', 'Endoglin', 'FGF2', 'Follistatin', 'G-CSF', 'GDF15', 'HE4', 'HGF', 'IL-6', 'IL-8', 'Kallikrein-6', 'Leptin', 'Mesothelin', 'Midkine', 'Myeloperoxidase', 'OPG', 'OPN', 'PAR', 'Prolactin', 'sEGFR', 'SHBG', 'sHER2/sEGFR2/sRbB2', 'TGf $\alpha$ ', 'Thrombospondin-2', 'TIMP-1', 'TIMP-2']  
Remaining CSV Files after eliminating top 5 Total Positive Cancers:  
['FeatureVectorCancerDet1.csv', 'FeatureVectorCancerDet2.csv', 'FeatureVectorCancerDet3.csv', 'FeatureVectorCancerDet4.csv', 'FeatureVectorCancerDet5.csv', 'FeatureVectorCancerDet6.csv', 'FeatureVectorCancerDet7.csv', 'FeatureVectorCancerDet8.csv', 'FeatureVectorCancerDet9.csv', 'FeatureVectorCancerDet10.csv', 'FeatureVectorCancerDet11.csv', 'FeatureVectorCancerDet12.csv', 'FeatureVectorCancerDet14.csv', 'FeatureVectorCancerDet15.csv', 'FeatureVectorCancerDet16.csv', 'FeatureVectorCancerDet17.csv', 'FeatureVectorCancerDet18.csv', 'FeatureVectorCancerDet19.csv', 'FeatureVectorCancerDet20.csv', 'FeatureVectorCancerDet21.csv', 'FeatureVectorCancerDet22.csv', 'FeatureVectorCancerDet23.csv', 'FeatureVectorCancerDet24.csv', 'FeatureVectorCancerDet25.csv', 'FeatureVectorCancerDet27.csv', 'FeatureVectorCancerDet28.csv', 'FeatureVectorCancerDet29.csv', 'FeatureVectorCancerDet30.csv', 'FeatureVectorCancerDet31.csv', 'FeatureVectorCancerDet33.csv', 'FeatureVectorCancerDet34.csv', 'FeatureVectorCancerDet36.csv', 'FeatureVectorCancerDet37.csv', 'FeatureVectorCancerDet38.csv', 'FeatureVectorCancerDet39.csv']  
The Significant Biomarkers: ['AFP', 'Angiopoietin-2', 'AXL', 'CA-125', 'CA 15-3', 'CA19-9', 'CD44', 'CEA', 'CYFRA 21-1', 'Endoglin', 'FGF2', 'Galectin-3', 'G-CSF', 'GDF15', 'HE4', 'HGF', 'IL-6', 'Leptin', 'Mesothelin', 'Midkine', 'Myeloperoxidase', 'OPG', 'OPN', 'PAR', 'Prolactin', 'sEGFR', 'SHBG', 'sHER2/sEGFR2/sRbB2', 'TGf $\alpha$ ', 'Thrombospondin-2', 'TIMP-1', 'TIMP-2']  
Eliminated least significant 5 biomarkers: ['Follistatin', 'NSE', 'sFas', 'sPECAM-1', 'DKK1']
```

15. Verify the least significant biomarkers with existing cancer biomarker literature. If any biomarker is already validated as a major biomarker in cancer detection for the given “**tumor_types**,” change the “**distance_threshold_offset**” parameter in Step 7 with another value and repeat the process.
16. If no major biomarkers are in the least significant biomarkers list, copy the Significant Biomarkers list.



```
Select_Biomarkers=['Angiopoietin-2', 'CA-125', 'CA 15-3', 'CEA', 'CYFRA 21-1', 'FGF2', 'G-CSF', 'HE4', 'HGF', 'PAR', 'sPECAM-1', 'Thrombospondin-2']  
Remaining CSV Files after eliminating top 5 Total Positive Cancers:  
['FeatureVectorCancerDet1.csv', 'FeatureVectorCancerDet2.csv', 'FeatureVectorCancerDet3.csv', 'FeatureVectorCancerDet4.csv', 'FeatureVectorCancerDet5.csv', 'FeatureVectorCancerDet6.csv', 'FeatureVectorCancerDet7.csv', 'FeatureVectorCancerDet8.csv', 'FeatureVectorCancerDet9.csv', 'FeatureVectorCancerDet10.csv', 'FeatureVectorCancerDet11.csv', 'FeatureVectorCancerDet12.csv', 'FeatureVectorCancerDet14.csv', 'FeatureVectorCancerDet15.csv', 'FeatureVectorCancerDet16.csv', 'FeatureVectorCancerDet17.csv', 'FeatureVectorCancerDet18.csv', 'FeatureVectorCancerDet19.csv', 'FeatureVectorCancerDet20.csv', 'FeatureVectorCancerDet21.csv', 'FeatureVectorCancerDet22.csv', 'FeatureVectorCancerDet23.csv', 'FeatureVectorCancerDet24.csv', 'FeatureVectorCancerDet25.csv', 'FeatureVectorCancerDet27.csv', 'FeatureVectorCancerDet28.csv', 'FeatureVectorCancerDet29.csv', 'FeatureVectorCancerDet30.csv', 'FeatureVectorCancerDet31.csv', 'FeatureVectorCancerDet33.csv', 'FeatureVectorCancerDet34.csv', 'FeatureVectorCancerDet36.csv', 'FeatureVectorCancerDet37.csv', 'FeatureVectorCancerDet38.csv', 'FeatureVectorCancerDet39.csv']  
The Significant Biomarkers: ['AFP', 'Angiopoietin-2', 'AXL', 'CA-125', 'CA 15-3', 'CA19-9', 'CD44', 'CEA', 'CYFRA 21-1', 'Endoglin', 'FGF2', 'Galectin-3', 'G-CSF', 'GDF15', 'HE4', 'HGF', 'IL-6', 'IL-8', 'Kallikrein-6', 'Leptin', 'Mesothelin', 'Midkine', 'Myeloperoxidase', 'OPG', 'OPN', 'PAR', 'Prolactin', 'sEGFR', 'SHBG', 'sHER2/sEGFR2/sRbB2', 'TGf $\alpha$ ', 'Thrombospondin-2', 'TIMP-1', 'TIMP-2']  
Eliminated least significant 5 biomarkers: ['Follistatin', 'NSE', 'sFas', 'sPECAM-1', 'DKK1']
```

Biomarker Selection Process

17. Paste the copied biomarkers list in “**Select_Biomarkers**”.
18. Enter a “**distance_threshold_offset**” value between 0 and 2.

```
jupyter Optimized Biomarker Selection Last Checkpoint: 21 hours ago
File Edit View Run Kernel Settings Help Trusted
JupyterLab Python 3 (ipykernel) ○
print(f"The Significant Biomarkers: {remaining_biomarkers}")

eliminated_biomarkers = [biomarker_file_map[file_name] for file_name, _ in top_files if file_name in biomarker_file_map]
print(f"Eliminated least significant {top_n_files_to_eliminate} biomarkers: {eliminated_biomarkers}")

# Execute the parts in sequence
part_1()
Normal_threshold = part_2()
biomarker_file_map, file_biomarker_map = part_3()
part_4(biomarker_file_map, file_biomarker_map)

# Run the function with user-defined values
Optimized_Biomarker_Selection(
    distance_threshold_offset=0.7,
    tumor_types={'normal': 554, 'breast': 156, 'colorectum': 291, 'esophagus': 34, 'liver': 33, 'lung': 78, 'ovary': 40, 'pancreas': 69, 'stomach': 50},
    max_patients=1802,
    normal_data_size=554,
    top_n_files_to_eliminate=5,
    cancer_types=['Breast', 'Colorectum', 'Esophagus', 'Liver', 'Lung', 'Ovary', 'Pancreas', 'Stomach'],
    Select_Biomarkers=['AFP', 'Angiopoietin-2', 'AXL', 'CA-125', 'CA 15-3', 'CA19-9', 'CD44', 'CEA', 'CYFRA 21-1', 'Endoglin', 'FGF2', 'Galectin-3', 'G-C'],
    clinical_data_path=r'D:\Cancer Detection folder\Biomarker Selection\Clinical cancer data.xlsx',
    feature_vector_path=r'D:\Cancer Detection folder\Biomarker Selection\FeatureVectorCancerDet.csv',
    output_folder_path=r'D:\Cancer Detection folder\Biomarker Selection',
    required_normal_positives=36 # User-defined required number of normal positives
)
2
```

19. Run the function and observe the Total Positive Cancers number again (Iteration. 2).

```
jupyter Optimized Biomarker Selection Last Checkpoint: 19 hours ago
File Edit View Run Kernel Settings Help Trusted
JupyterLab Python 3 (ipykernel) ○
1-Dim Hole Max Life Range Average Birth (1D) Average Death (1D) \
0 0.150208 0.727339 0.777248
1 0.152941 0.727713 0.778923
2 0.152730 0.726848 0.778186
3 0.145399 0.724122 0.775472
4 0.146079 0.719524 0.769463
5 0.151641 0.726687 0.778010
6 0.154016 0.727799 0.779603
7 0.172995 0.721548 0.775183
8 0.159407 0.723177 0.774687

Regression Difference Percentage Distance %(1)
0 0.519086 0.644740
1 0.828860 0.925743
2 0.909420 1.087417
3 1.094389 1.253983
4 1.144593 1.481474
5 0.723247 0.791001
6 1.509318 1.119891
7 1.188847 1.424509
8 1.571227 1.550974

Final 'Distance %(1)' Threshold (k_2): 1.344740
Final Regression Difference Threshold (k_1): 0.2430
Total Positive Cancers: 265
Normal Threshold (k_2): 1.3447398809460138
```

20. If the Total Positive Cancers number is not satisfactory, then continue from Step 15 to 19 again.
21. Repeat this process until you get a satisfactory number of Total Positive Cancers.

22. If a satisfactory number of Total Positive Cancers is obtained for a certain biomarker set, and further elimination of biomarkers does not improve the Total Positive Cancers number, that biomarker set can be considered the final significant biomarker set.

Iteration No.	Regression Difference Threshold (k_1)	Total Cancer Positives	Distance %(1) Threshold (k_2) (Distance %(1) + distance_threshold_offset)		Biomarkers eliminated	top_n_files_to_eliminate
			Distance %(1)	distance_threshold_offset		
1	0.682	142	0.830194	0.7	'Follistatin', 'NSE', 'sFas', 'DKK1', 'sEGFR'	5
2	0.243	265	0.64474	0.7	'TIMP-2', 'Galectin-3', 'TGF α ', 'TIMP-1', 'SHBG'	5
3	0.454	332	0.59581	0.4	'AXL', 'Leptin', 'Midkine', 'Kallikrein-6', 'IL-8'	5
4	0.248	317	0.710615	0.8	'OPG', 'Prolactin', 'AFP', 'Mesothelin', 'Endoglin'	5
5	0.723	353	0.900462	0.6	'Myeloperoxidase', 'CD44', 'OPN', 'GDF15', 'sHER2/sEGFR2/sErbB2'	5
6	0.662	448	1.053634	0.8	'IL-6', 'CA19-9'	2
7	-	480	-	-	-	-

23. Use the default parameters For **distance_threshold_offset** and **top_n_files_to_eliminate**, from **Table 1** in the respective iterations to get the example significant biomarkers set: 'Angiopoietin-2', 'CA-125', 'CA 15-3', 'CEA', 'CYFRA 21-1', 'FGF2', 'G-CSF', 'HE4', 'HGF', 'PAR', 'sPECAM-1', 'Thrombospondin-2'.

Biomarker Selection Process

jupyter Optimized Biomarker Selection Last Checkpoint: 21 hours ago

File Edit View Run Kernel Settings Help Trusted JupyterLab Python 3 (ipykernel)

```
remaining_biomarkers = [biomarker_file_map[file_name] for file_name in remaining_file_names if file_name in biomarker_file_map]
print(f"The Significant Biomarkers: {remaining_biomarkers}")

eliminated_biomarkers = [biomarker_file_map[file_name] for file_name, _ in top_files if file_name in biomarker_file_map]
print(f"Eliminated least significant {top_n_files_to_eliminate} biomarkers: {eliminated_biomarkers}")

# Execute the parts in sequence
part_1()
Normal_threshold = part_2()
biomarker_file_map, file_biomarker_map = part_3()
part_4(biomarker_file_map, file_biomarker_map)

# Run the function with user-defined values
Optimized_Biomarker_Selection(
    distance_threshold_offset=0.8,
    tumor_types={'normal': 554, 'breast': 156, 'colorectum': 291, 'esophagus': 34, 'liver': 33, 'lung': 78, 'ovary': 40, 'pancreas': 69, 'stomach': 50},
    max_patients=1802,
    normal_data_size=554,
    top_n_files_to_eliminate=2,
    cancer_types=['Breast', 'Colorectum', 'Esophagus', 'Liver', 'Lung', 'Ovary', 'Pancreas', 'Stomach'],
    Select_Biomarkers=['Angiopoietin-2', 'CA-125', 'CA 15-3', 'CEA', 'CYFRA 21-1', 'FGF2', 'G-CSF', 'HE4', 'HGF', 'PAR', 'sPECAM-1', 'Thrombospondin-2'],
    clinical_data_path=r'D:\Cancer Detection folder\Biomarker Selection\Clinical cancer data.xlsx',
    feature_vector_path="D:/Cancer Detection folder/Biomarker Selection/FeatureVectorCancerDet.csv",
    output_folder="D:/Cancer Detection folder/Biomarker Selection",
    required_normal_positives=36 # User-defined required number of normal positives
)
```

jupyter Optimized Biomarker Selection Last Checkpoint: 21 hours ago

File Edit View Run Kernel Settings Help Trusted JupyterLab Python 3 (ipykernel)

	1-Dim Hole Max Life Range	Average Birth (1D)	Average Death (1D)
0	0.079376	0.714055	0.757050
1	0.088306	0.736295	0.781624
2	0.088962	0.739001	0.782889
3	0.081022	0.751810	0.797381
4	0.099900	0.755252	0.803019
5	0.073667	0.756284	0.798887
6	0.068588	0.768394	0.809247
7	0.073682	0.743884	0.788008
8	0.091454	0.777279	0.828486

	Regression Difference Percentage	Distance %(1)
0	0.585122	1.232344
1	1.341531	4.349512
2	1.290672	4.819940
3	1.751234	6.632627
4	1.672438	8.287415
5	1.468677	6.400702
6	2.196203	8.063345
7	1.826669	5.772096
8	1.836260	9.829385

Final 'Distance %(1)' Threshold (k_2): 2.032344

Final Regression Difference Threshold (k_1): 0.4440

Total Positive Cancers: 488

Normal Threshold (k_2): 2.0323436735107046

24. Final Significant Biomarkers set: 'Angiopoietin-2', 'CA-125', 'CA 15-3', 'CEA', 'CYFRA 21-1', 'FGF2', 'G-CSF', 'HE4', 'HGF', 'PAR', 'sPECAM-1', 'Thrombospondin-2'.

Threshold Determination

25. Enter the same inputs for feature_vector_path, Selection/FeatureVectorCancerDet.csv", normal_data_size, cancer_types and tumor_types in the **get_thresholds** function.

The screenshot shows a Jupyter Notebook interface with a code cell containing Python code. The code defines a function `get_thresholds` which imports pandas, numpy, and LinearRegression from sklearn. The function takes parameters: `feature_vector_path`, `normal_data_size`, `distance_threshold_offset`, `cancer_types`, `tumor_types`, and `required_normal_positives`. It calculates thresholds for `Normal_distance_threshold`, `Regression Difference Threshold`, and `Total Positive Cancers`. It then loads the feature vector data from the CSV file. The `construct_linear_regression_model` function prepares the data for regression by selecting the first `normal_data_size` rows. A red circle highlights the `get_thresholds` function definition.

```
[3]: # Determination of the final parameters for the cancer detection model

[6]: import pandas as pd
      import numpy as np
      from sklearn.linear_model import LinearRegression

def get_thresholds(
    feature_vector_path, # Path to the feature vector CSV file
    normal_data_size, # Number of normal patients in the dataset
    distance_threshold_offset, # Offset for the distance threshold
    cancer_types, # List of cancer types to calculate total positive cancers
    tumor_types, # Dictionary of tumor types and their row limits
    required_normal_positives # User-defined required number of normal positives
):
    """
    Calculate and return the thresholds:
    - Normal_distance_threshold (k_2)
    - Regression Difference Threshold (k_1)
    - Total Positive Cancers
    """

    # Load the feature vector data
    new_data = pd.read_csv(feature_vector_path, header=0)

    def construct_linear_regression_model():
        # Prepare the data for regression
        data = new_data.iloc[:normal_data_size]
```

26. Run the function **get_thresholds**.

The screenshot shows a Jupyter Notebook interface with a code cell containing Python code. The code demonstrates the usage of the `get_thresholds` function with specific input values: `feature_vector_path` set to "D:/Cancer Detection folder/Biomarker Selection/FeatureVectorCancerDet.csv", `normal_data_size` set to 554, `cancer_types` set to ['Breast', 'Colorectum', 'Esophagus', 'Liver', 'Lung', 'Ovary', 'Pancreas', 'Stomach'], and `tumor_types` set to {'normal': 554}. The code then initializes variables for the best result, loops through `distance_threshold_offset` from -1 to 2 with an increment of 0.1, and prints the results for each iteration. A red box highlights the input parameter definitions.

```
# Example usage
feature_vector_path = "D:/Cancer Detection folder/Biomarker Selection/FeatureVectorCancerDet.csv"
normal_data_size = 554
cancer_types = ['Breast', 'Colorectum', 'Esophagus', 'Liver', 'Lung', 'Ovary', 'Pancreas', 'Stomach']
tumor_types = {'normal': 554}

# Initialize variables to store the best result
best_total_positive_cancers = 0
best_Normal_distance_threshold = 0
best_regression_threshold = 0
best_normal_1_dim_hole_max_life_range = 0

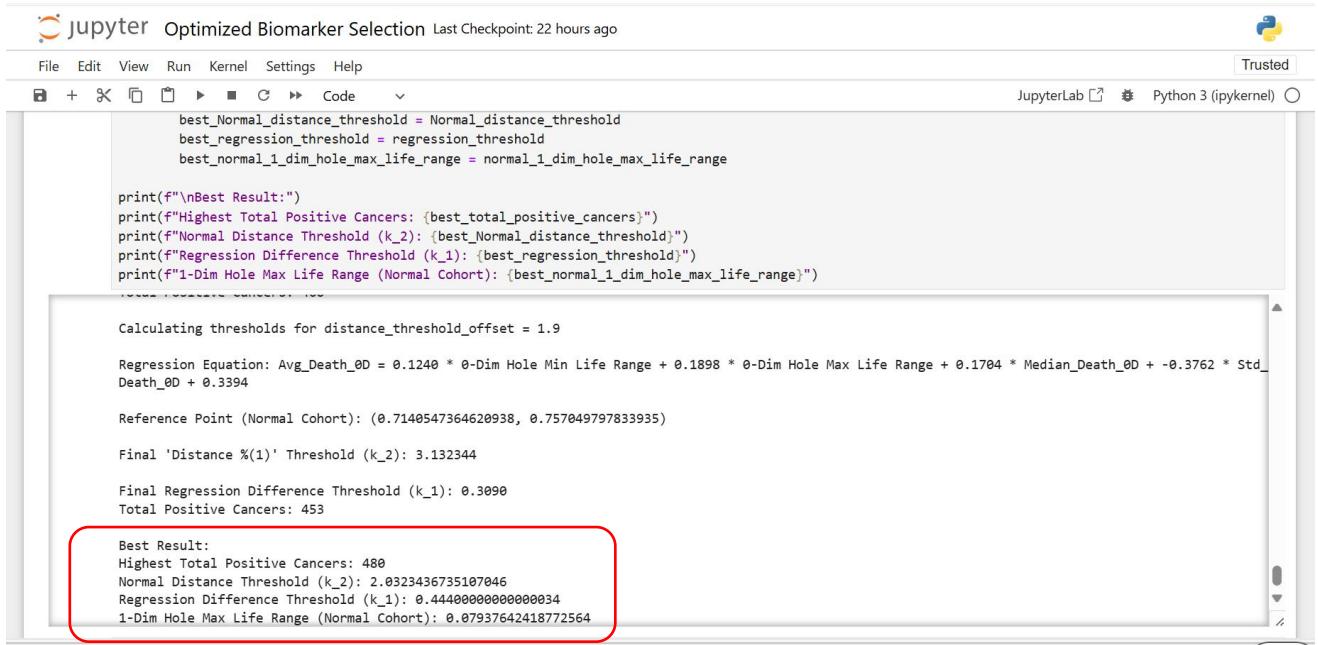
# User-defined required number of normal positives
required_normal_positives = 36 # Example: User-defined value

# Loop through distance_threshold_offset from -1 to 2 with an increment of 0.1:
for distance_threshold_offset in np.arange(-1, 2, 0.1):
    print(f"\nCalculating thresholds for distance_threshold_offset = {distance_threshold_offset:.1f}")
    Normal_distance_threshold, regression_threshold, total_positive_cancers, normal_1_dim_hole_max_life_range = get_thresholds(
        feature_vector_path, normal_data_size, distance_threshold_offset, cancer_types, tumor_types, required_normal_positives
    )

    if total_positive_cancers is not None and total_positive_cancers > best_total_positive_cancers:
        best_total_positive_cancers = total_positive_cancers
        best_Normal_distance_threshold = Normal_distance_threshold
        best_regression_threshold = regression_threshold
        best_normal_1_dim_hole_max_life_range = normal_1_dim_hole_max_life_range
```

Threshold Determination

27. The thresholds to maximize cancer detection sensitivity for the selected biomarker set will be obtained.



Jupyter Optimized Biomarker Selection Last Checkpoint: 22 hours ago Trusted

File Edit View Run Kernel Settings Help JupyterLab Python 3 (ipykernel)

```
best_Normal_distance_threshold = Normal_distance_threshold
best_regression_threshold = regression_threshold
best_normal_1_dim_hole_max_life_range = normal_1_dim_hole_max_life_range

print(f"\nBest Result:")
print(f"Highest Total Positive Cancers: {best_total_positive_cancers}")
print(f"Normal Distance Threshold (k_2): {best_Normal_distance_threshold}")
print(f"Regression Difference Threshold (k_1): {best_regression_threshold}")
print(f"1-Dim Hole Max Life Range (Normal Cohort): {best_normal_1_dim_hole_max_life_range}")

Calculating thresholds for distance_threshold_offset = 1.9

Regression Equation: Avg_Death_0D = 0.1240 * 0-Dim Hole Min Life Range + 0.1898 * 0-Dim Hole Max Life Range + 0.1704 * Median_Death_0D + -0.3762 * Std_Death_0D + 0.3394

Reference Point (Normal Cohort): (0.7140547364620938, 0.757049797833935)

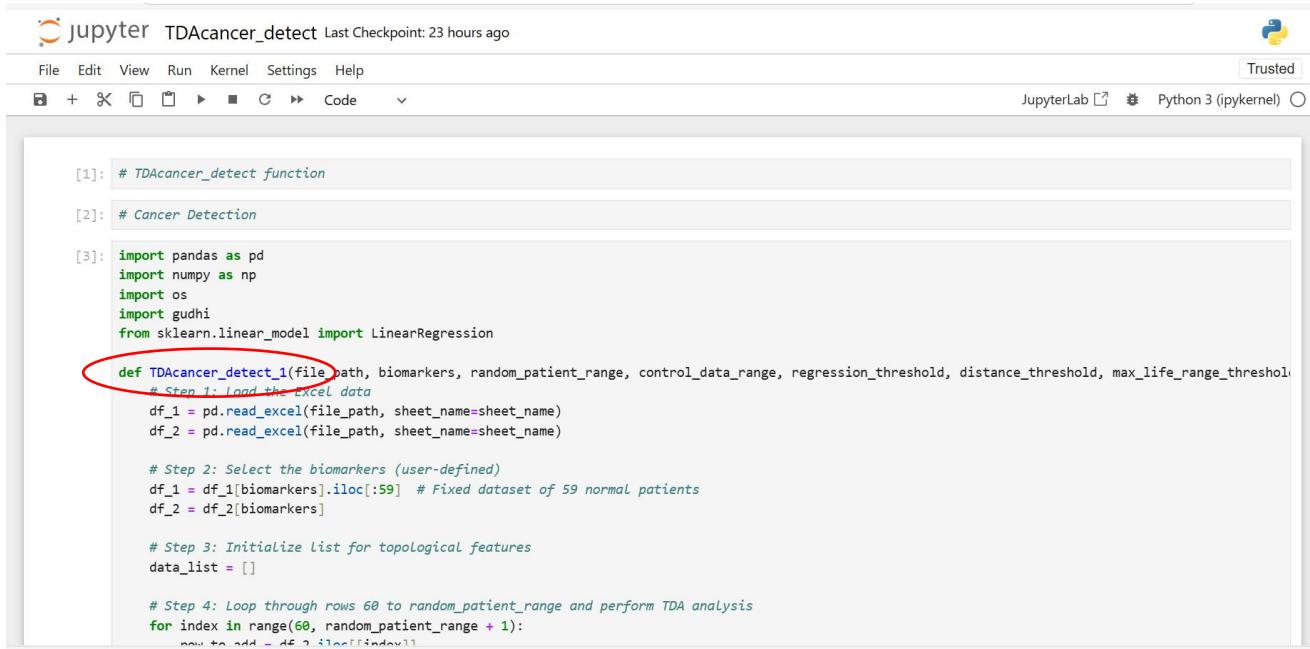
Final 'Distance %(1)' Threshold (k_2): 3.132344

Final Regression Difference Threshold (k_1): 0.3090
Total Positive Cancers: 453
```

Best Result:
Highest Total Positive Cancers: 480
Normal Distance Threshold (k_2): 2.0323436735107046
Regression Difference Threshold (k_1): 0.44400000000000034
1-Dim Hole Max Life Range (Normal Cohort): 0.07937642418772564

Cancer Detection

28. Load “*TDAcancer_detect.ipynb*” in a Jupyter Notebook.



```
[1]: # TDACancer_detect function
[2]: # Cancer Detection
[3]: import pandas as pd
import numpy as np
import os
import gudhi
from sklearn.linear_model import LinearRegression

def TDACancer_detect_1(file_path, biomarkers, random_patient_range, control_data_range, regression_threshold, distance_threshold, max_life_range_threshold):
    # Step 1: Load the excel data
    df_1 = pd.read_excel(file_path, sheet_name=sheet_name)
    df_2 = pd.read_excel(file_path, sheet_name=sheet_name)

    # Step 2: Select the biomarkers (user-defined)
    df_1 = df_1[biomarkers].iloc[:59] # Fixed dataset of 59 normal patients
    df_2 = df_2[biomarkers]

    # Step 3: Initialize List for topological features
    data_list = []

    # Step 4: Loop through rows 60 to random_patient_range and perform TDA analysis
    for index in range(60, random_patient_range + 1):
        now_to_add = df_2.iloc[index].values
        data_list.append(now_to_add)
```

29. Enter the required inputs to the function ***TDACancer_detect_1***. The inputs’ descriptions are given below.

- **file_path:** Path to the file containing the Excel file ‘**Clinical cancer data.xlsx**’.
- **sheet_name:** Input the patients’ data sheet ‘Normal and Cancer’ from the provided Excel Data File ‘**Clinical cancer data.xlsx**’.
- **biomarkers:** Select the biomarkers obtained from the ***Optimized_Biomarker_Selection*** algorithm (Step 24).
- **random_patient_range:** Specify the total number of patients in the dataset.
- **control_data_range:** Provide the number of healthy individuals to use as training samples.
- **tumor_types:** List the tumor types present in the dataset.

30. Enter the parameters **k_1** and **k_2** obtained from the ***get_thresholds*** algorithm.

31. After entering **k_1** and **k_2**, enter the parameters **k_3** and **k_4**. Select **k_3** and **k_4** such that both have the same distance from the value “**1-Dim Hole Max Life Range (Normal Cohort)**” obtained in the ***get_thresholds*** algorithm. Additionally, in selecting **k_3** and **k_4**, maintain the “Normal Positive” (False Cancer Positives) to ensure specificity is at least over 95%.

Note: For the example file “**Clinical cancer data.xlsx**”, the user must provide **file_path**. The other inputs are set as default. The inputs are case-sensitive, so refer to the example inputs and ensure that your inputs match the given case exactly.

Cancer Detection

32. Run the function **TDAcancer_detect_1**.



```
jupyter TDAcancer_detect Last Checkpoint: 22 hours ago
File Edit View Run Kernel Settings Help Trusted
JupyterLab Python 3 (ipykernel) ○

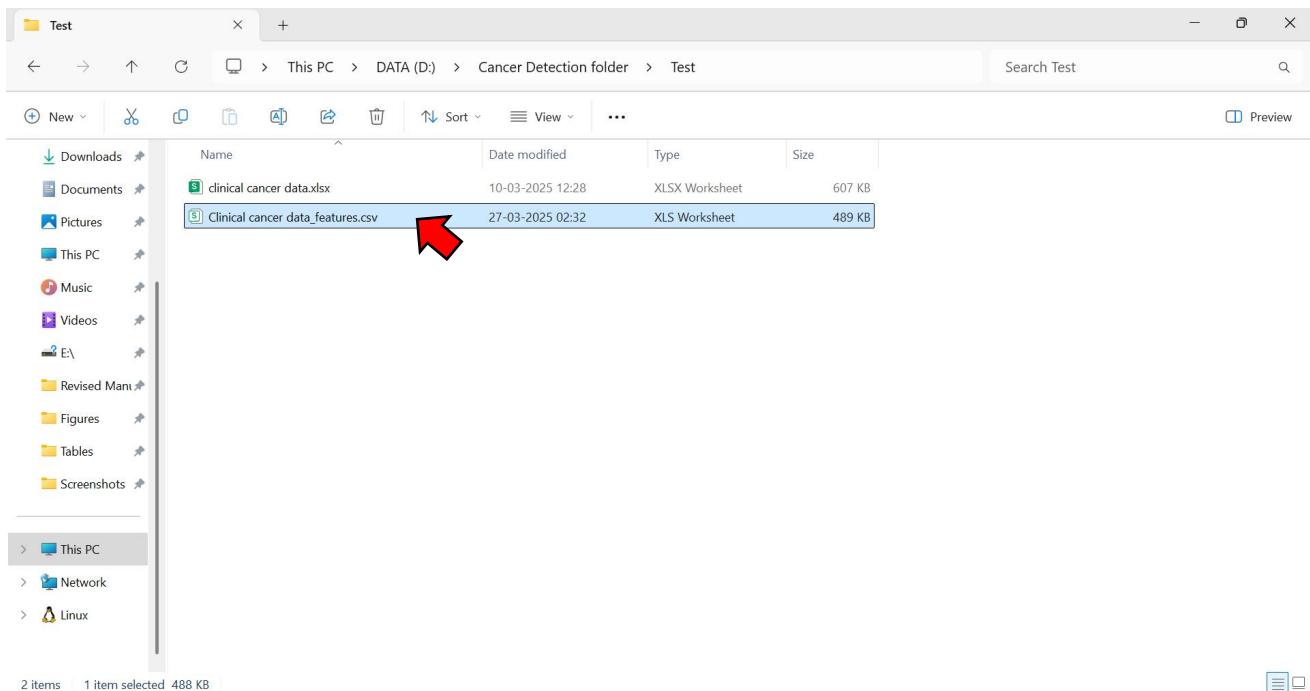
results["Total Positive Cancers"] = sum(positive_counts.values) - positive_counts.get('Normal', 0)

return results

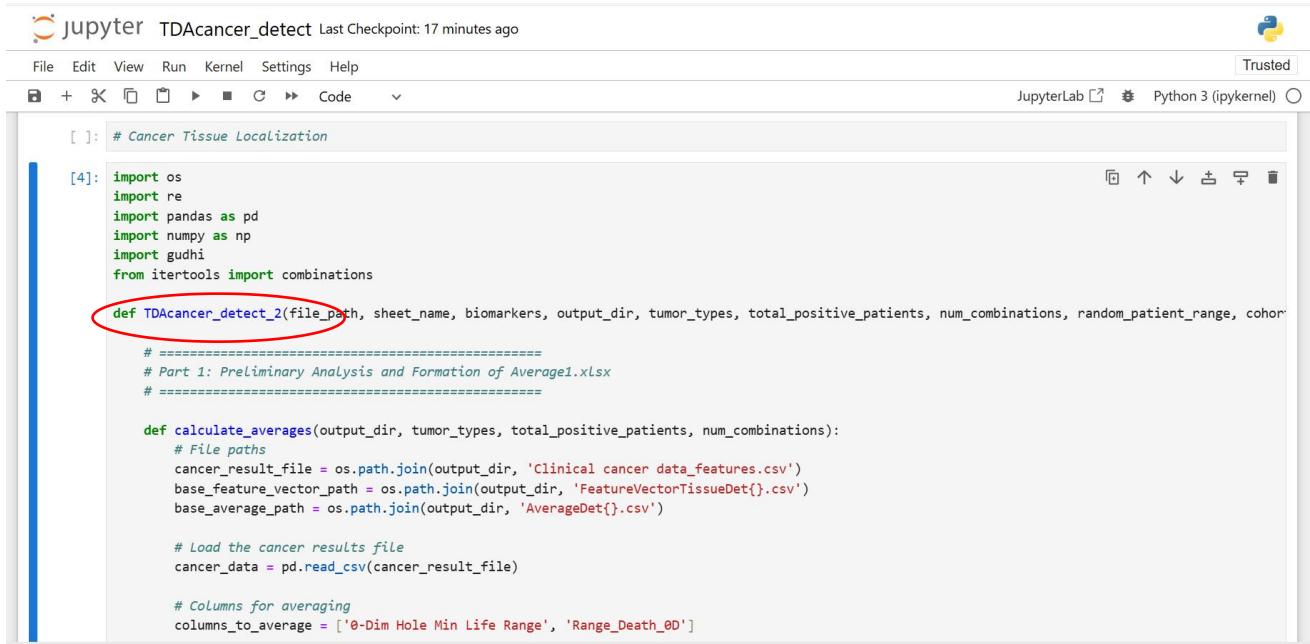
# Example usage:
file_path = r'D:\Cancer Detection folder\Test\Clinical cancer data.xlsx'
sheet_name = 'Normal and Cancer'
# User-defined variables
biomarkers = ['Angiopoietin-2', 'CA-125', 'CA 15-3', 'CEA', 'CYFRA 21-1', 'FGF2', 'G-CSF', 'HE4', 'HGF', 'PAR', 'sPECAM-1', 'Thrombospondin-2']
random_patient_range = 1802 # User-defined range
control_data_range = 554 # User-defined control data range
k_1 = 0.444 # User-defined regression threshold
k_2 = 2.03 # User-defined distance threshold
k_3 = 0.112 # User-defined upper threshold for 1D hole max Life range
k_4 = 0.047 # User-defined Lower threshold for 1D hole max Life range
tumor_types = ['Normal', 'Breast', 'Colorectum', 'Esophagus', 'Liver', 'Lung', 'Ovary', 'Pancreas', 'Stomach'] # User-defined tumor types

# Run the pipeline
results = TDAcancer_detect_1(
    file_path, biomarkers, random_patient_range, control_data_range,
    k_1, k_2, k_3, k_4, tumor_types
)
print(results)
{'Normal Positive': 36, 'Breast Positive': 77, 'Colorectum Positive': 182, 'Esophagus Positive': 25, 'Liver Positive': 29, 'Lung Positive': 47, 'Ovary Positive': 41, 'Pancreas Positive': 42, 'Stomach Positive': 55, 'Total Positive Cancers': 498}
```

33. A CSV file “**Clinical cancer data_features.csv**” will be generated in the given “**file_path**.” This file contains the cancer detection results (Positive or Negative) for each individual patient.



Cancer Tissue Localization



The screenshot shows a Jupyter Notebook interface with the title "jupyter TDACancer_detect Last Checkpoint: 17 minutes ago". The menu bar includes File, Edit, View, Run, Kernel, Settings, Help, and a Trusted button. Below the menu is a toolbar with various icons. The code cell [4] contains Python code for "Cancer Tissue Localization". A red oval highlights the function definition `def TDACancer_detect_2(file_path, sheet_name, biomarkers, output_dir, tumor_types, total_positive_patients, num_combinations, random_patient_range, cohorts_to_analyze)`. The code imports os, re, pandas, numpy, gudhi, and itertools. It defines the `TDACancer_detect_2` function and a nested `calculate_averages` function. The `calculate_averages` function reads a CSV file and performs averaging operations.

```
[4]: import os
import re
import pandas as pd
import numpy as np
import gudhi
from itertools import combinations

def TDACancer_detect_2(file_path, sheet_name, biomarkers, output_dir, tumor_types, total_positive_patients, num_combinations, random_patient_range, cohorts_to_analyze):

    # =====#
    # Part 1: Preliminary Analysis and Formation of Average1.xlsx
    # =====#

    def calculate_averages(output_dir, tumor_types, total_positive_patients, num_combinations):
        # File paths
        cancer_result_file = os.path.join(output_dir, 'Clinical_cancer_data_features.csv')
        base_feature_vector_path = os.path.join(output_dir, 'FeatureVectorTissueDet{}.csv')
        base_average_path = os.path.join(output_dir, 'AverageDet{}.csv')

        # Load the cancer results file
        cancer_data = pd.read_csv(cancer_result_file)

        # Columns for averaging
        columns_to_average = ['0-Dim Hole Min Life Range', 'Range_Death_0D']

34. Enter the required inputs to the function TDACancer_detect_2. The inputs' descriptions are given below.

- file_path: Path to the file containing the Excel file 'Clinical cancer data.xlsx'.
- sheet_name: Input the patients' data sheet 'Normal and Cancer' from the provided Excel Data File 'Clinical cancer data.xlsx'.
- biomarkers: Select the biomarkers.
- output_dir: Path to store the result files.
- tumor_types: List the tumor types present in the dataset along with the number of patients to use as training samples. The sample numbers should be selected from true negative healthy individuals, and for cancer tumors, from true positives obtained from TDACancer_detect_1.
- total_positive_patients: List the cancer types present in the dataset along with the number of cancer-positive patients identified by TDACancer_detect_1.
- cohorts_to_analyze: List the cancer types present in the dataset.
- num_combinations: Provide the number of biomarker combinations user want to analyze.
- random_patient_range: Specify the total number of patients in the dataset.



*Note: For the example file "Clinical cancer data.xlsx", the user must provide file_path and output_dir. The user should set the same input and output file paths for the functions TDACancer_detect_1 and TDACancer_detect_2 because the files used in these functions are interconnected. The other inputs are set as default. The inputs are case-sensitive, so refer to the example inputs and ensure that your inputs match the given case exactly.


```

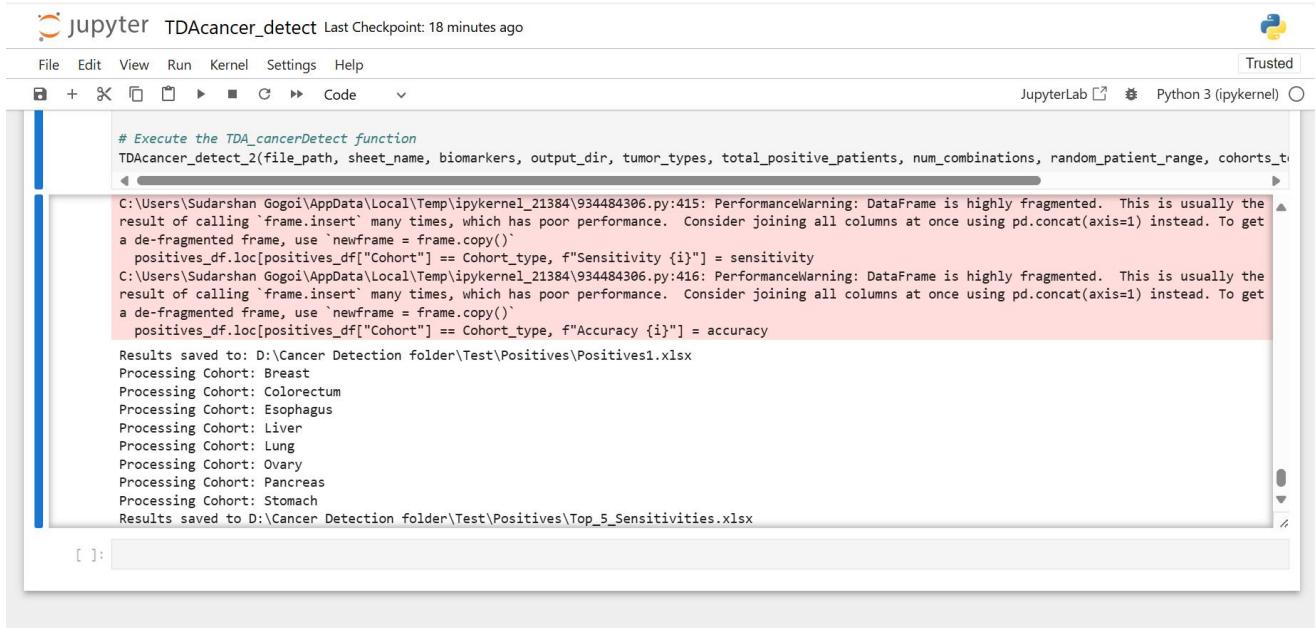
Cancer Tissue Localization

35. Run the function **TDAcancer_detect_2**.



```
# Example usage
file_path = r'D:\Cancer Detection folder\Test\Clinical cancer data.xlsx'
sheet_name = 'Normal and Cancer'
biomarkers = ['Angiopoietin-2', 'CA-125', 'CA 15-3', 'CEA', 'CYFRA 21-1', 'FGF2', 'G-CSF', 'HE4', 'HGF', 'PAR', 'sPECAM-1', 'Thrombospondin-2']
output_dir = r"D:\Cancer Detection folder\Test"
# User-defined inputs
tumor_types = {
    'normal': 528,
    'breast': 58,
    'colon': 137,
    'esophagus': 18,
    'liver': 22,
    'lung': 35,
    'ovary': 31,
    'pancreas': 32,
    'stomach': 41
}
total_positive_patients = {
    "Breast": 77, "Colon": 182, "Esophagus": 25, "Liver": 29,
    "Lung": 47, "Ovary": 41, "Pancreas": 42, "Stomach": 55
}
cohorts_to_analyze = ['Breast', 'Colon', 'Esophagus', 'Liver', 'Lung', 'Ovary', 'Pancreas', 'Stomach']
num_combinations = 220 # User-defined number of combinations
random_patient_range = 1802 # User-defined random patient range
# Execute the TDA_cancerDetect function
TDAcancer_detect_2(file_path, sheet_name, biomarkers, output_dir, tumor_types, total_positive_patients, num_combinations, random_patient_range, cohorts_to_analyze)
```

36. The result files will be saved in the given **output_dir**.

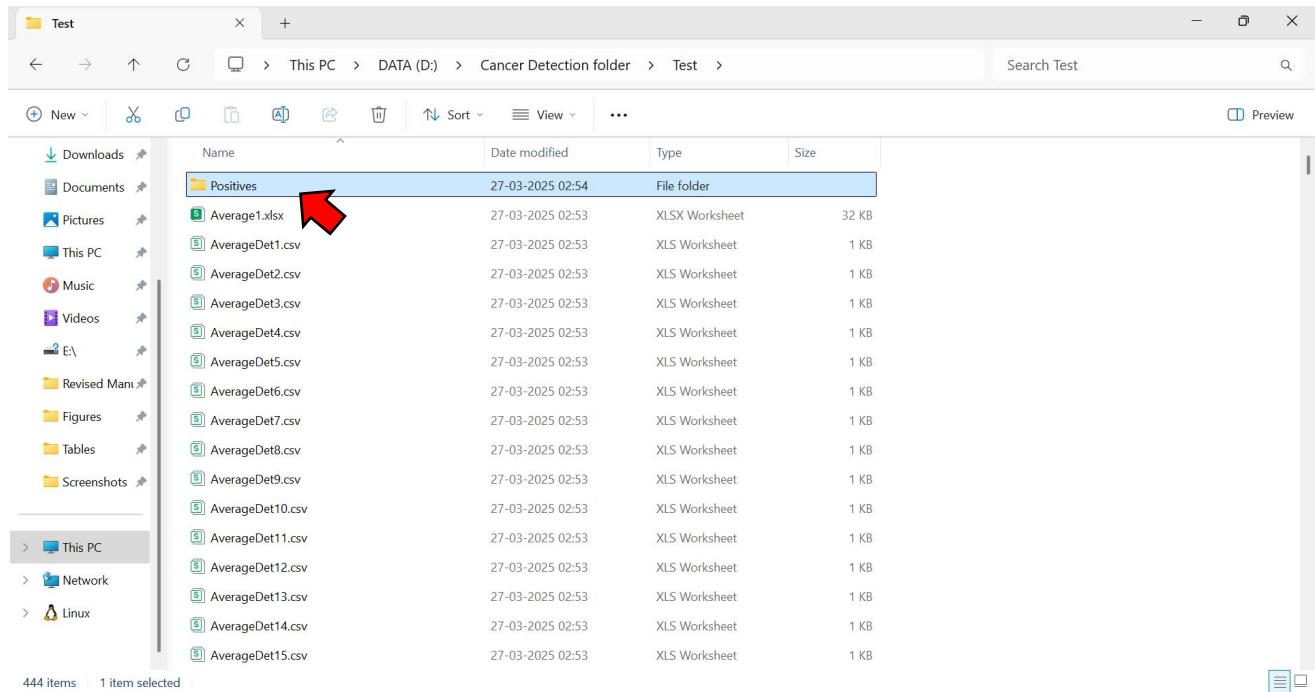


```
# Execute the TDA_cancerDetect function
TDAcancer_detect_2(file_path, sheet_name, biomarkers, output_dir, tumor_types, total_positive_patients, num_combinations, random_patient_range, cohorts_to_analyze)

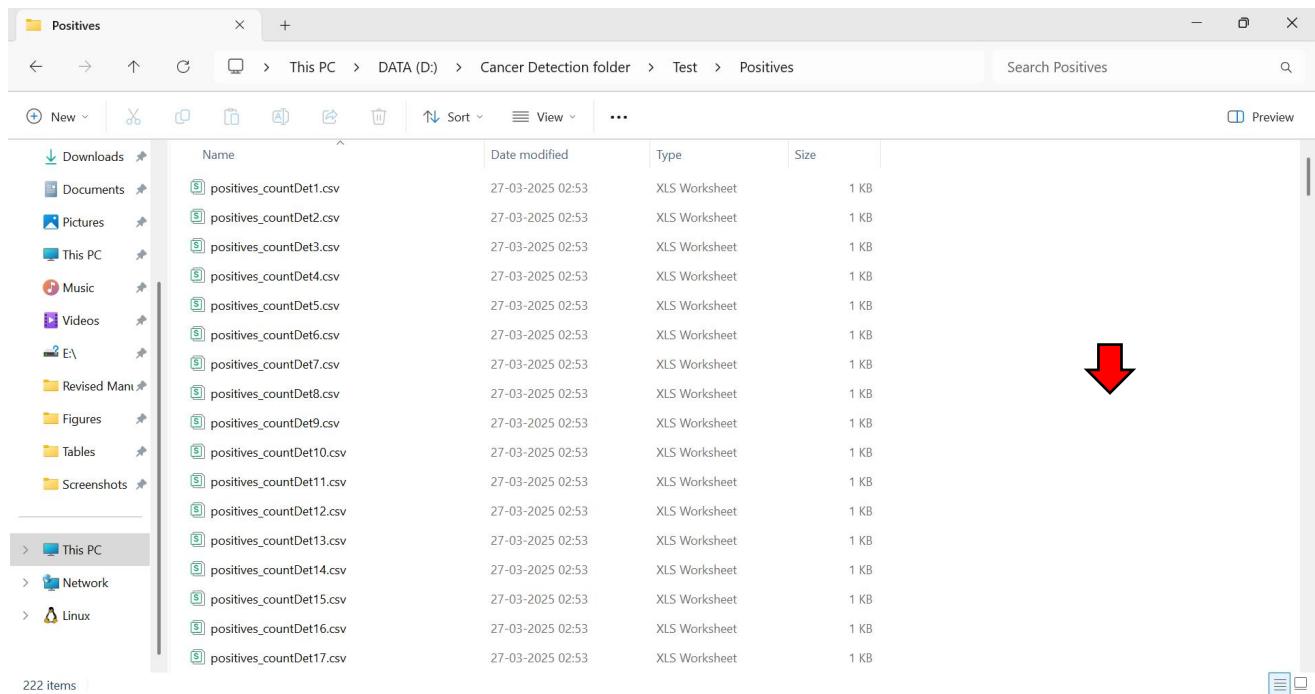
C:\Users\Sudarshan Gogoi\AppData\Local\Temp\ipykernel_21384\934484306.py:415: PerformanceWarning: DataFrame is highly fragmented. This is usually the result of calling `frame.insert` many times, which has poor performance. Consider joining all columns at once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = frame.copy()`
    positives_df.loc[positives_df["Cohort"] == Cohort_type, f"Sensitivity {i}"] = sensitivity
C:\Users\Sudarshan Gogoi\AppData\Local\Temp\ipykernel_21384\934484306.py:416: PerformanceWarning: DataFrame is highly fragmented. This is usually the result of calling `frame.insert` many times, which has poor performance. Consider joining all columns at once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = frame.copy()`
    positives_df.loc[positives_df["Cohort"] == Cohort_type, f"Accuracy {i}"] = accuracy
Results saved to: D:\Cancer Detection folder\Test\Positives\Positives1.xlsx
Processing Cohort: Breast
Processing Cohort: Colon
Processing Cohort: Esophagus
Processing Cohort: Liver
Processing Cohort: Lung
Processing Cohort: Ovary
Processing Cohort: Pancreas
Processing Cohort: Stomach
Results saved to: D:\Cancer Detection folder\Test\Positives\Top_5_Sensitivities.xlsx
```

Cancer Tissue Localization

37. Open the **output_dir**.
38. Open the folder “**Positives**”

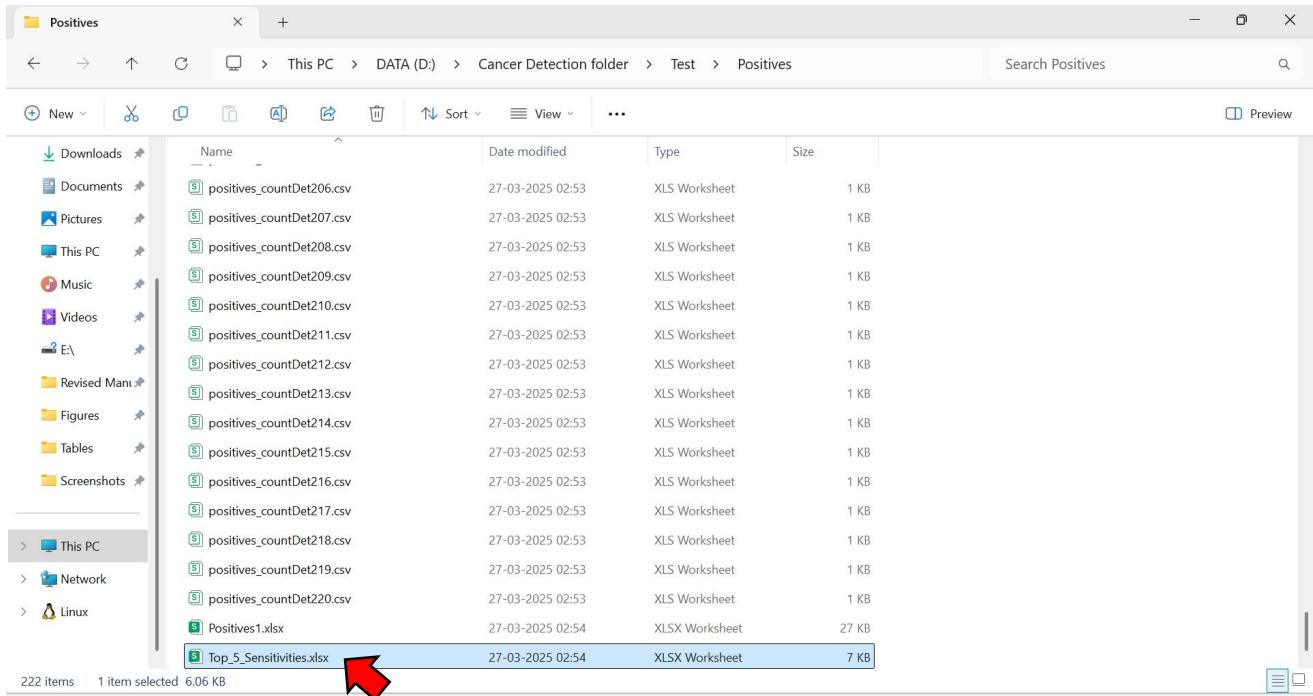


39. Scroll down to the last file.



Cancer Tissue Localization

40. Open Top_5_Sensitivities.xlsx file



41. Tissue localization sensitivity and accuracy for different biomarkers will be obtained.

The screenshot shows an Excel spreadsheet with the following details:

- Sheet: Sheet1
- Table Title: Biomarkers
- Table Structure:

	Cohort	Sensitivity	Accuracy	Biomarker	Sensitivity	Accuracy	Biomarker	Sensitivity	Accuracy	Biomarker	Sensitivity	Accuracy	Biomarker	Sensitivity	Accuracy	Biomarker	Sensitivity	Accuracy	Biomarker			
1	Breast	0.415584	0.700201	[CEA]	'FGI	0.376623	0.710262	[CYFRA 21]	0.363636	0.734406	[CA 15-3']	0.350649	0.708249	[FGF2]	'G	0.324675	0.653924	[FGF2]	'G-CSF'	sPECAM-1'		
2	Colorectal	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA			
3	Esophagus	0.708333	0.762575	[FGF2]	'H'	0.583333	0.722334	[G-CSF]	'I'	0.583333	0.810865	[G-CSF]	'I'	0.583333	0.790744	[HGF]	'PA	0.541667	0.818913	[Angiopoietin-2]	'CA 15-3'	'HE4'
4	Liver	0.586207	0.810865	[Angiopo	0.551724	0.790744	[HGF]	'sPI	0.517241	0.768612	[CA 15-3']	0.482759	0.782696	[G-CSF]	'P	0.482759	0.756539	[HE4]	'PAR'	'Thrombospondin-1'		
5	Lung	0.659574	0.615694	[CA 15-3']	0.659574	0.609658	[CYFRA 21]	0.638298	0.607646	[CEA]	'CYI	0.617021	0.603622	[Angiopo	0.617021	0.617706	[CYFRA 21-1]	'HE4'	'PAR'			
6	Ovary	0.902439	0.853119	[CA-125']	0.902439	0.859155	[CA-125']	0.878049	0.875252	[Angiopo	0.878049	0.802817	[CA-125']	0.878049	0.867203	[CA-125']	'FGF2'	'G-CSF'				
7	Pancreas	0.571429	0.839034	[CA 15-3']	0.547619	0.720322	[Angiopo	0.547619	0.734406	[Angiopo	0.52381	0.804829	[Angiopo	0.52381	0.814889	[CEA]	'HE4'	'PAR'				
8	Stomach	0.763636	0.748491	[CEA]	'CYI	0.636364	0.696177	[CEA']	0.618182	0.704225	[CA-125']	0.581818	0.726358	[CEA']	'CYI	0.563636	0.732394	[CYFRA 21-1]	'FGF2'	sPECAM-1'		
- Bottom Status Bar: 100%