CancerVision: Advanced Breast Cancer Prediction with Deep Learning

A MINI PROJECT REPORT

SUBMITTED BY

LAWANYA M – 412720104026

TRISHA G– 412720104046

DEEPIKA J – 412720104701

SHALINI S – 412720104040

IN PARTIAL FULFILLMENT FOR THE AWARD OF THE DEGREE OF

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO
1	**INTRODUCTION** 1.1 PROJECT OVERVIEW 1.2 PURPOSE	 8 8 8
2	**IDEATION & PROPOSED SOLUTION** 2.1 PROBLEM STATEMENT DEFINITION 2.2 EMPATHY MAP CANVAS 2.3 IDEATION & BRAINSTORMING 2.4 PROPOSED SOLUTION	 11 12 12 12 13
3	**REQUIREMENT ANALYSIS** 3.1 FUNCTIONAL REQUIREMENT 3.2 NON-FUNCTIONAL REQUIREMENTS	 17 17
4	**PROJECT DESIGN** 4.1 DATA FLOW DIAGRAMS 4.2 SOLUTION & TECHNICAL ARCHITECTURE 4.3 USER STORIES	 18 19 19
5	**CODING & SOLUTIONING** 5.1 Feature 1 5.2 Feature 2 5.3 Database Schema (if Applicable)	 19 20
6	**RESULTS** 6.1 Performance Metrics	 22 22 35
7	**ADVANTAGES & DISADVANTAGES**	
8	**CONCLUSION**	37
9	**FUTURE SCOPE**	
10	**APPENDIX** Source Code	

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

Introduction:

CancerVision is an innovative project aimed at developing an advanced breast cancer prediction system using deep learning techniques. Breast cancer is a significant health concern globally, and early detection plays a crucial role in improving patient outcomes. The project leverages the power of deep learning algorithms to analyze medical imaging data and provide accurate predictions of breast cancer presence and progression.

Objectives:

The primary objectives of CancerVision are as follows:

a. Develop a deep learning model capable of analyzing breast imaging data, including mammograms, ultrasounds, and MRIs.

b. Train the model using a large dataset of annotated breast cancer images to learn the complex patterns and features indicative of cancerous tissue.

c. Create a user-friendly interface that allows healthcare professionals to input breast imaging data and receive predictions on the presence of breast cancer.

d. Enhance the accuracy and reliability of breast cancer prediction compared to existing methods, leading to earlier detection and more effective treatment.

Methodology:

The CancerVision project follows a systematic methodology involving several key steps:

a. Data Collection: Gather a comprehensive dataset of breast imaging data, including mammograms, ultrasounds, and MRIs, along with corresponding clinical information and pathology reports.

b. Data Preprocessing: Clean and preprocess the imaging data, including image normalization, resizing, and augmentation techniques, to ensure consistency and enhance the learning process.

c. Model Architecture Design: Develop a deep learning architecture, such as convolutional neural networks (CNNs), that can effectively extract features from the breast imaging data and make accurate predictions.

d. Training and Validation: Train the deep learning model using the annotated dataset, optimizing the model's parameters through iterations to improve prediction performance. Validate the model using cross-validation techniques to assess its generalizability.

e. User Interface Development: Create a user-friendly interface that allows healthcare professionals to input breast imaging data easily and obtain predictions in real-time.

f. Evaluation and Testing: Evaluate the performance of CancerVision by comparing its predictions with ground truth labels and existing diagnostic methods. Conduct rigorous testing to ensure the model's reliability and accuracy.

g. Refinement and Optimization: Continuously refine and optimize the CancerVision system based on feedback from healthcare professionals and further improvements in deep learning techniques.

Expected Outcomes:

The CancerVision project aims to achieve the following outcomes:

a. Accurate Breast Cancer Prediction: Develop a highly accurate deep learning model capable of predicting the presence and progression of breast cancer with a high degree of sensitivity and specificity.

b. Early Detection and Improved Treatment: Enable earlier detection of breast cancer, leading to timely interventions, improved treatment outcomes, and potentially saving lives.

c. User-Friendly Interface: Create an intuitive and user-friendly interface that allows healthcare professionals to easily access and interpret predictions, facilitating clinical decision-making.

d. Enhanced Healthcare Delivery: Contribute to the advancement of healthcare by integrating state-of-the-art deep learning technology into breast cancer diagnosis, potentially reducing workload and improving diagnostic accuracy.

1.2 PURPOSE

The purpose of the CancerVision:

Advanced Breast Cancer Prediction with Deep Learning project is to address the critical need for accurate and early detection of breast cancer using deep learning techniques. The project aims to revolutionize breast cancer diagnosis by leveraging the power of deep learning algorithms to analyze breast imaging data, such as mammograms, ultrasounds, and MRIs. By doing so, it seeks to achieve the following purposes:

Improve Detection Accuracy:

Breast cancer can be challenging to detect, especially in its early stages. CancerVision aims to develop a deep learning model that can analyze complex patterns and features in breast imaging data, leading to improved accuracy in identifying cancerous tissue. By enhancing the detection process, the project aims to reduce false negatives and false positives, resulting in more reliable diagnostic outcomes.

Enable Early Diagnosis:

Early detection is crucial in improving breast cancer treatment outcomes. CancerVision seeks to identify breast cancer at its earliest stages when it is most treatable. By utilizing deep learning algorithms, the project aims to detect subtle signs of cancer that might be missed by human observers, enabling timely interventions and potentially saving lives.

Enhance Clinical Decision-making:

The project aims to provide healthcare professionals with a powerful tool for breast cancer prediction. By integrating CancerVision into clinical practice, healthcare providers can make more informed decisions regarding patient care, treatment plans, and follow-up strategies. The system's predictions can serve as an additional resource to assist radiologists and oncologists in their diagnostic evaluations.

Facilitate Personalized Treatment:

CancerVision has the potential to contribute to personalized medicine in breast cancer treatment. By accurately predicting the presence and progression of breast cancer, the project can help tailor treatment plans to individual patients' specific needs. This may include selecting appropriate therapies, determining the extent of surgical interventions, and assessing the efficacy of treatment options over time.

Contribute to Medical Research:

The CancerVision project generates valuable insights and data that can contribute to medical research and further advancements in breast cancer diagnosis and treatment. The project's findings, methodologies, and techniques can be shared with the scientific community, potentially leading to improvements in deep learning algorithms, data collection, and diagnostic protocols.

CHAPTER 2

IDEATION & PROPOSED SOLUTION

2.1 PROBLEM STATEMENT DEFINITION

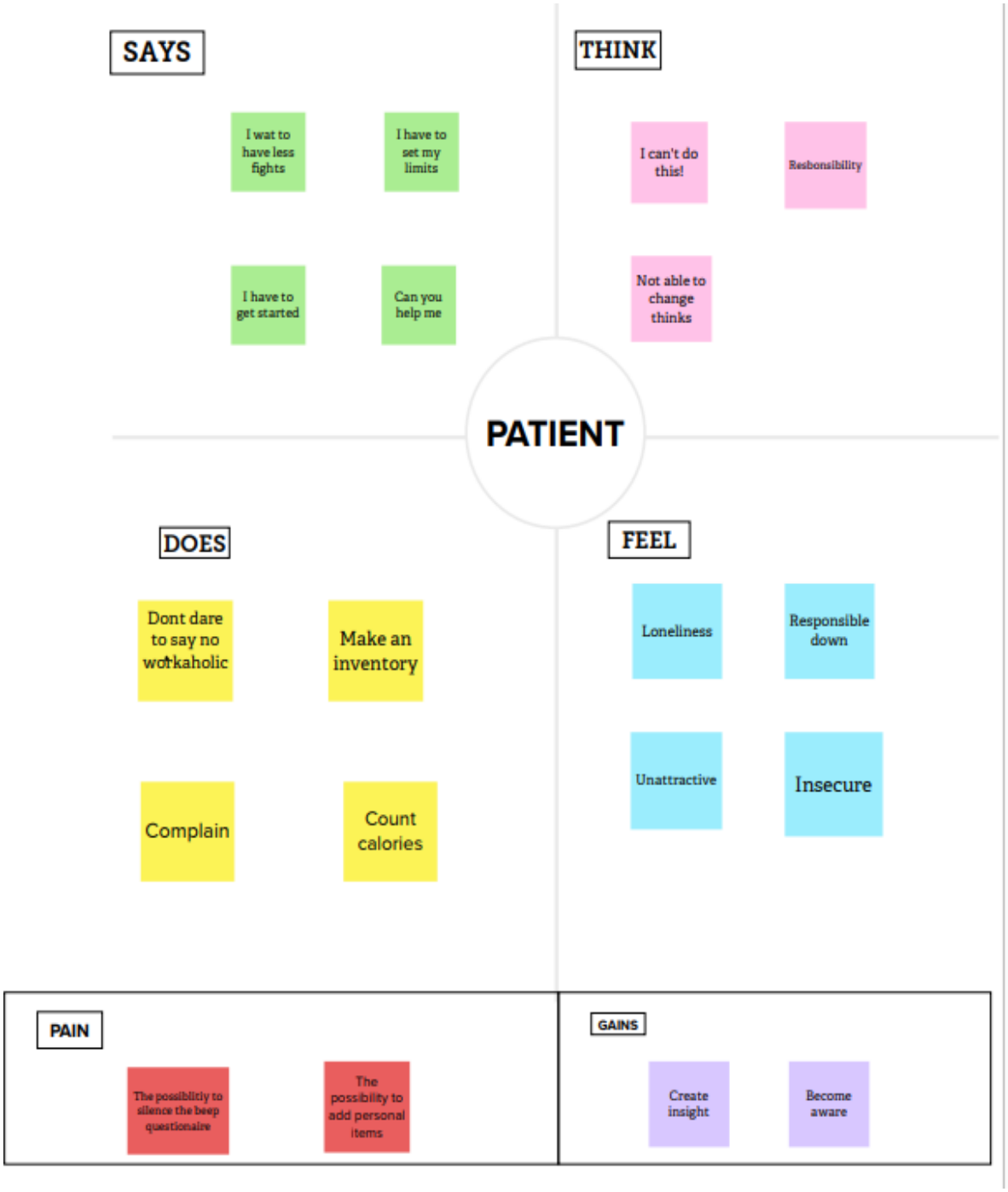
The problem at hand is to develop a deep learning-based predictive model that can accurately identify and predict advanced breast cancer in patients. This model will utilize advanced machine learning algorithms, specifically deep neural networks, to analyze complex patterns and relationships within large-scale breast cancer datasets. By incorporating a wide range of patient-related factors, including medical history, clinical data, imaging results, and genetic information, the model aims to improve the accuracy of advanced breast cancer prediction compared to existing methods.

The outcome of this research will be a sophisticated deep learning model that can assist medical professionals in identifying patients at high risk of advanced breast cancer. Such a predictive tool will enable early intervention, personalized treatment plans, and improved patient outcomes. Additionally, the model's interpretability will be explored, allowing clinicians to understand the underlying factors contributing to its predictions and aiding in clinical decision-making.

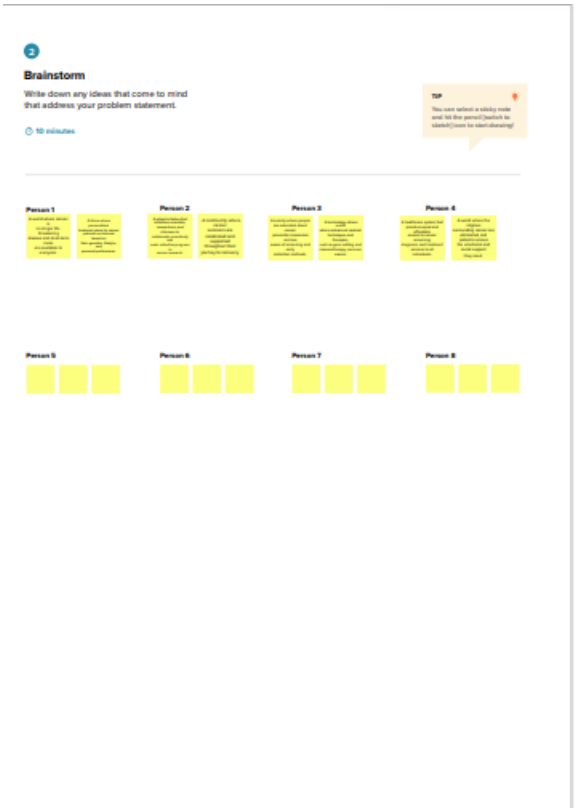
PROBLEM STATEMENT 1:

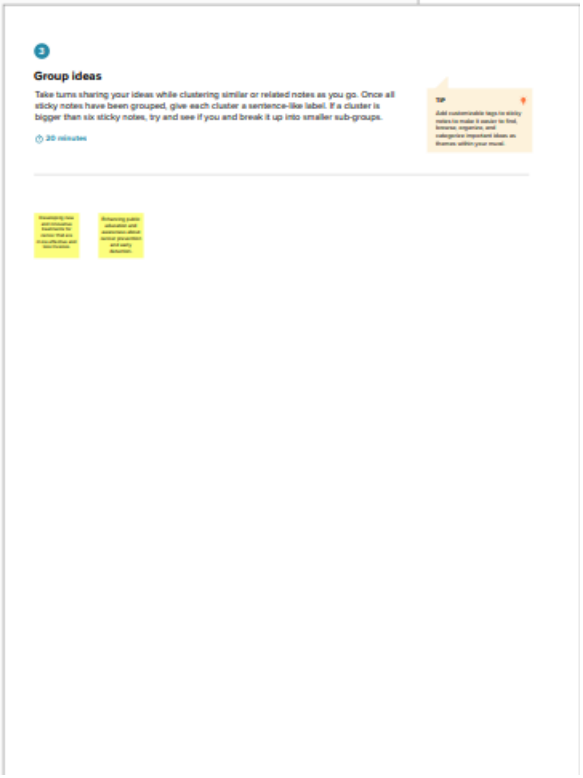
PROBLEM STATEMENT 2:

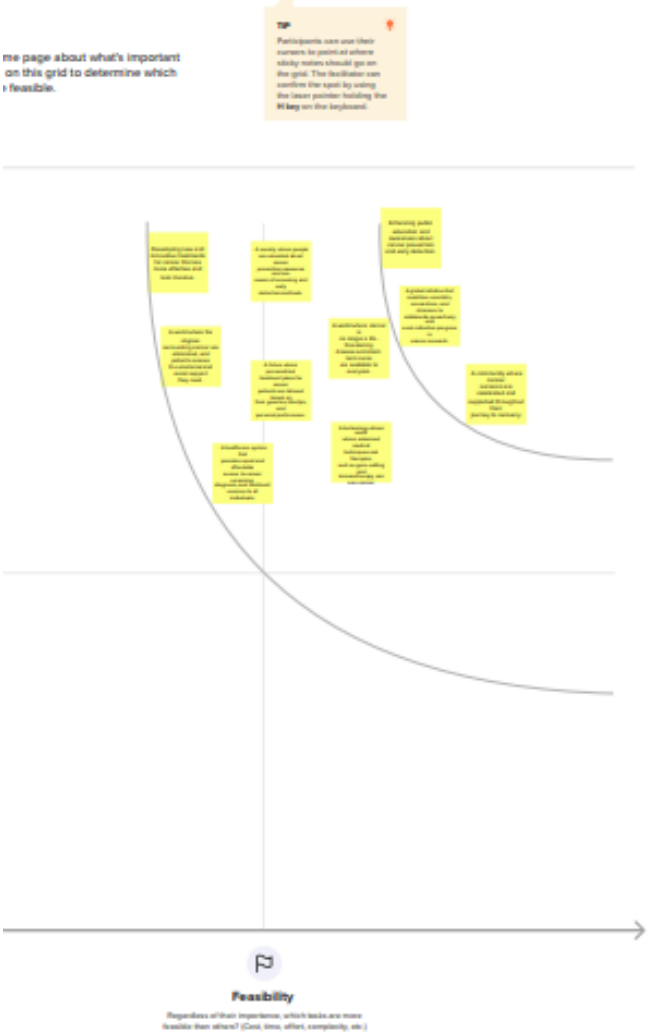
2.2 EMPATHY MAP CANVAS

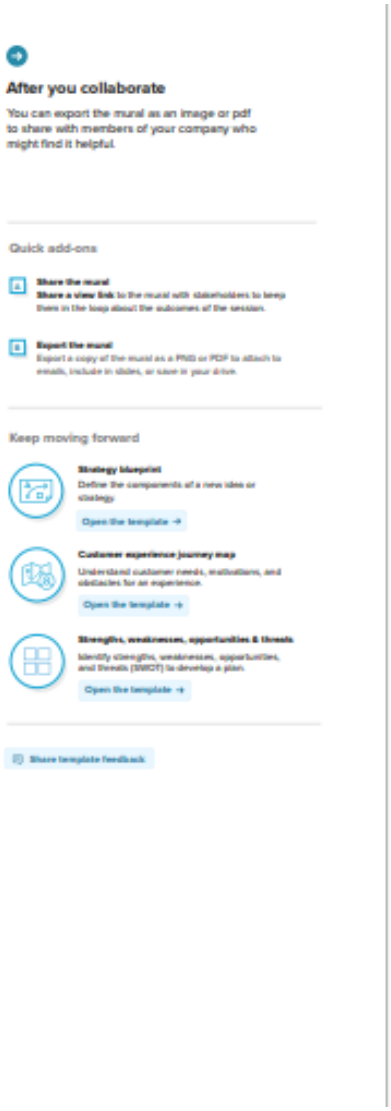


2.3 IDEATION BRAINSTORMING


Brainstorm
Write down any ideas that come to mind that address your problem statement.
10 minutes
TIP
Person 1
Person 2
Person 3
Person 4
Person 5
Person 6
Person 7
Person 8


Group ideas
Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.
20 minutes
TIP


TIP
Feasibility


After you collaborate
You can export the mural as an image or pdf to share with members of your company who might find it helpful.
Quick add-ons
Share the mural
Export the mural
Keep moving forward
Strategy blueprint
Define the components of a new idea or strategy.
Open the template
Customer experience journey map
Understand customer needs, motivations, and obstacles for an experience.
Open the template
Strengths, weaknesses, opportunities & threats
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
Open the template
Share template feedback

2.4 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Deep Learning, a subset of Machine Learning, has shown promising results in the field of medical image analysis, including breast cancer detection. By leveraging large datasets of mammography images, Deep Learning models can learn to accurately identify the features and patterns associated with advanced breast cancer. CancerVision could use a large dataset of mammography images to train a Deep Learning model that can accurately detect advanced breast cancer. The system could be integrated with existing healthcare infrastructure to enable efficient and accurate diagnosis of advanced breast cancer, thus improving patient outcomes and reducing healthcare costs. CancerVision has the potential to revolutionize the detection and treatment of advanced breast cancer, providing patients with a more effective and less invasive means of diagnosis and treatment.
2.	Idea / Solution description	The model is designed to analyze a wide range of factors including medical history, lifestyle, and genetic data to generate a highly personalized risk assessment. The model is designed to identify patterns and relationships between different factors and use this information to make highly accurate predictions about breast cancer risk. To use the CancerVision model, patients provide their medical history, lifestyle information, and genetic data. The model then analyzes this information and generates a personalized risk assessment report. The report includes a detailed analysis of the patient's risk factors and provides recommendations for reducing the risk of developing breast cancer. The CancerVision model has been extensively tested and has been shown to be highly accurate in predicting breast cancer risk. CancerVision has the potential to save countless lives and improve the health outcomes of women around the world.

3.	Novelty / Uniqueness	One potential approach for predicting advanced breast cancer using deep learning is to use a convolution neural network (CNN) to analyze medical images, such as mammograms or MRIs, and identify patterns or features that are associated with advanced breast cancer. The CNN can be trained using a large dataset of medical images and corresponding clinical data, including patient age, family history, tumor size, and other relevant factors. Another approach is to use deep learning to analyze genomic data from breast cancer patients, identifying genetic mutations. This could involve training a deep neural network to analyze gene expression data, DNA sequencing data, or other types of genomic data and predict the likelihood of advanced breast cancer based on the presence or absence of specific genetic mutations or patterns.
4.	Social Impact / Customer Satisfaction	Early detection: Deep learning algorithms can analyze vast amounts of medical data and identify patterns that human experts might miss. By using CancerVision to detect advanced breast cancer at an earlier stage, patients can receive more effective treatments, and their chances of survival can improve. Accessibility: In many parts of the world, access to high-quality medical care is limited. CancerVision, which uses machine learning algorithms to analyze patient data, can help bring advanced breast cancer detection to patients in remote areas who might not have access to specialist physicians or diagnostic equipment. Affordability: Cancer treatment can be expensive, and patients in lower-income countries might not be able to afford the cost of advanced breast cancer detection. CancerVision, which can be integrated into existing medical infrastructure, could provide a more affordable alternative to traditional screening methods. Improved decision-making: The use of deep learning algorithms to analyze patient data could help physicians make more informed decisions about cancer treatment. By providing accurate predictions of disease progression, CancerVision could help doctors tailor treatments to individual patients, improving their chances of successful treatment.
5.	Business Model (Revenue Model)	Data collection: CancerVision collects a large amount of medical data related to breast cancer, including patient records, mammogram images, and biopsy results. This data is used to train their deep learning algorithms to accurately predict breast cancer. Algorithm development: CancerVision's team of data scientists and machine learning experts develop and refine the deep learning algorithms that power their predictive models. These algorithms are constantly

		updated and improved as new data becomes available. Model deployment: CancerVision's predictive models are deployed in hospitals and clinics, where they can be used by oncologists and radiologists to assist in the diagnosis of advanced breast cancer. Revenue generation: CancerVision generates revenue through licensing their predictive models to hospitals and clinics.They may also charge for consulting services, such as helping medical professionals interpret the results of the predictive models and develop treatment plans for patients. Overall, CancerVision's business model is focused on providing healthcare professionals with cutting-edge technology to improve the diagnosis and treatment of advanced breast cancer.
6.	Scalability of the Solution	Big data technologies: To scale the solution, CancerVision can use big data technologies such as Hadoop, Spark, and NoSQL databases. These technologies can handle large volumes of data and enable CancerVision to process the data faster and more efficiently. Distributed computing: Distributed computing can enable CancerVision to process large amounts of data in parallel across multiple computing nodes. Cloud computing: Cloud computing can provide a scalable and cost-effective solution for CancerVision. Automation: Automation can help CancerVision streamline its data processing and analysis workflows, reduce manual errors, and improve scalability. Collaborations: Collaboration with healthcare providers, hospitals, and clinics can help CancerVision collect more data and improve the accuracy of its predictions. API-based architecture: This architecture can enable easy integration of CancerVision's solution with electronic health record systems and other healthcare systems.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FUNCTIONAL REQUIREMENT

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Data collection and pre processing	The deep learning system should be able to collect and pre-process large volumes of data from multiple sources, including mammograms, MRIs, ultrasound scans, pathology reports, and patient medical histories.
FR-2	Feature extraction and selection	The system should be able to extract relevant features from the data, such as lesion size, shape, texture, and margins. It should also select the most important features that can be used for accurate prediction
FR-3	Model training and validation	The deep learning system should be able to train and validate a predictive model using the extracted features. This model should be able to accurately predict the risk of advanced breast cancer based on the patient's clinical data.
FR-4	User interface	The system should have a user-friendly interface that allows medical professionals to input patient data, view the model output, and interpret the results. The interface should also manage the storage and retrieval of data securely.
FR-5	Performance and accuracy	The system should be able to achieve high levels of accuracy and performance, with minimal false positives or false negatives.
FR-6	Integration with electronic health records	The system should be able to seamlessly integrate with electronic health records (EHRs) to access patient data and provide real-time predictions and recommendations to medical professionals.

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Data collection and pre processing	The deep learning system should be able to collect and pre-process large volumes of data from multiple sources, including mammograms, MRIs, ultrasound scans, pathology reports, and patient medical histories.
FR-2	Feature extraction and selection	The system should be able to extract relevant features from the data, such as lesion size, shape, texture, and margins. It should also select the most important features that can be used for accurate prediction
FR-3	Model training and validation	The deep learning system should be able to train and validate a predictive model using the extracted features. This model should be able to accurately predict the risk of advanced breast cancer based on the patient's clinical data.
FR-4	User interface	The system should have a user-friendly interface that allows medical professionals to input patient data, view the model output, and interpret the results. The interface should also manage the storage and retrieval of data securely.
FR-5	Performance and accuracy	The system should be able to achieve high levels of accuracy and performance, with minimal false positives or false negatives.
FR-6	Integration with electronic health records	The system should be able to seamlessly integrate with electronic health records (EHRs) to access patient data and provide real-time predictions and recommendations to medical professionals.

3.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional Requirements:

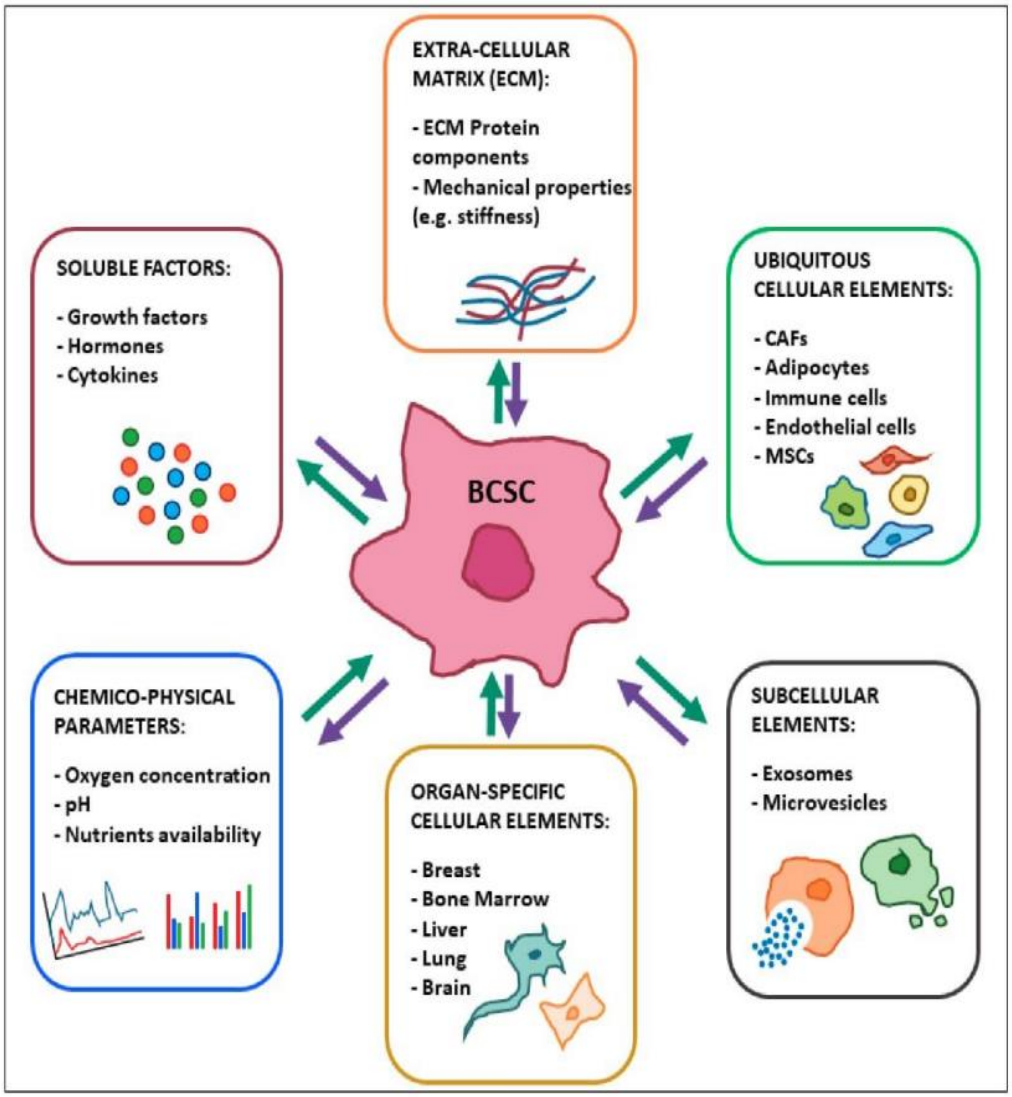
Following are the non-functional requirements of the proposed solution.

FR No.	Non- Functional requirements	Description
FR-1	Usability	The system should be easy to use and navigate for healthcare professionals. This means that it should have an intuitive user interface and be designed with user experience in mind.
FR-2	Security	The system should be designed with security in mind to prevent unauthorized access and protect sensitive patient data.
FR-3	Reliability	The system should be reliable and consistent in its predictions. This means that it should be able to produce the same results when given the same input data.
FR-4	Performance	The system should be fast and efficient in making predictions. This means that it should be optimized to run on hardware with high computational power.
FR-5	Accuracy	The system should have a high degree of accuracy in predicting the presence or absence of advanced breast cancer. This means that the model should be tested and validated on a large and diverse dataset to ensure that it is accurate.
FR-6	Scalability	The system should be able to handle large volumes of data and be able to scale up as needed to meet increasing demand.

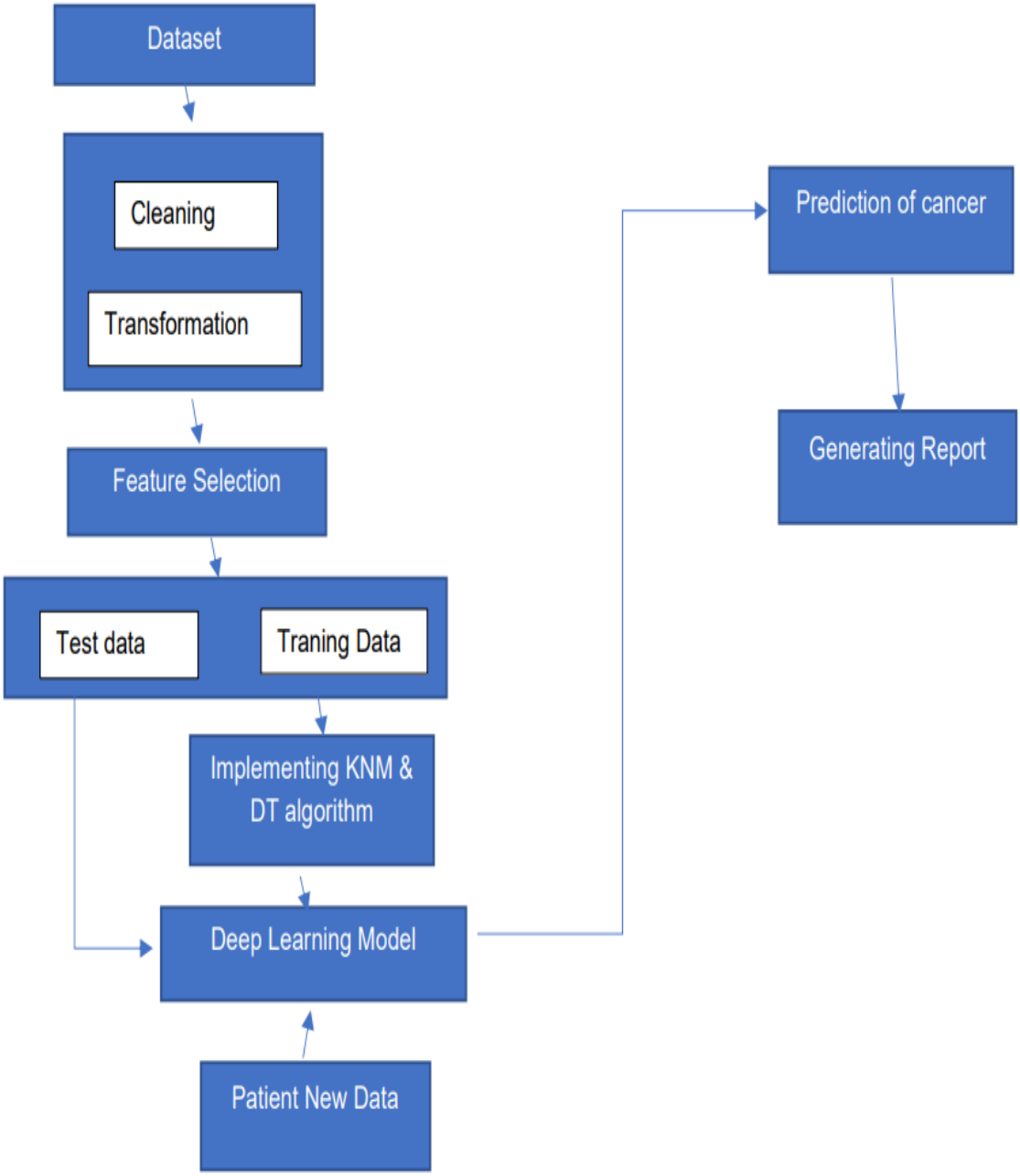
CHAPTER 4

PROJECT DESIGN

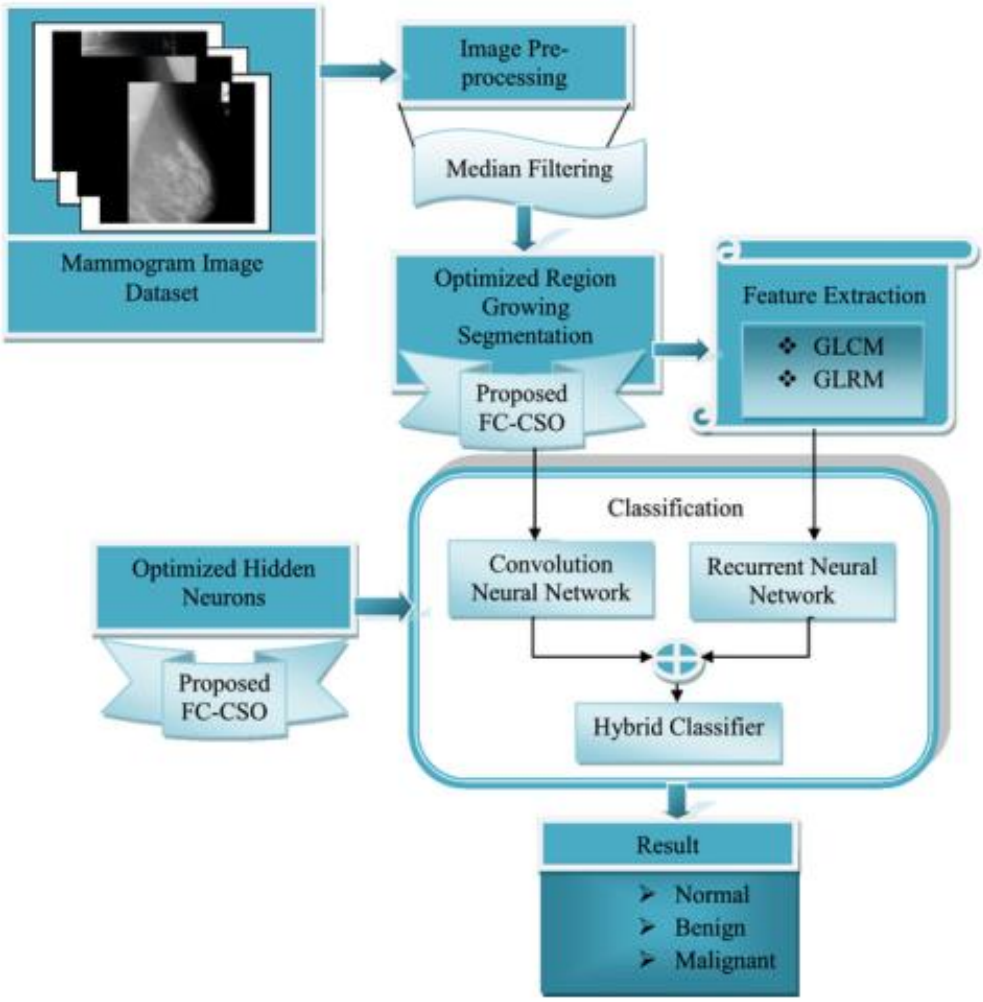
4.1 DATA FLOW DIAGRAMS



FLOW DIAGRAM:



4.2 SOLUTION & TECHNICAL ARCHITECTURE



4.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Tasks	Acceptance criteria	Priority	Team Member
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password	I can access my account / dashboard	High	Lawanya
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Lawanya
		USN-3	As a user, I can register for the application through Google	I can register & access the dashboard with Google Login	Low	Trisha
		USN-4	As a user, I can register for the application through Gmail		Medium	Trisha
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Shalini
	Dashboard	USN-6	If breast cancer is diagnosed, a woman will usually undergo treatment such as surgery, radiation therapy, chemotherapy, or targeted therapy. Treatment will depend on the type and stage of the cancer and may involve a combination of approaches		High	Deepika
Customer (web user)	Issues or changes		Other diagnostic tests, such as breast MRI, may also be used to investigate any concerning changes in breast tissue		High	Deepika
Administractor	Treatments		Emotional support can help women cope with the physical and psychological effects of breast cancer and its treatment			

CHAPTER 5

CODING & SOLUTIONING

5.1 FEATURE

Deep learning algorithms: CancerVision leverages state-of-the-art deep learning algorithms to analyze medical imaging data, such as mammograms and breast MRI scans. These algorithms can automatically extract relevant features and patterns from the images, enabling accurate predictions.

Early detection: One of the primary goals of CancerVision is early detection of breast cancer. By analyzing the imaging data, the system can identify subtle signs and indicators of cancerous cells at an early stage. Early detection is crucial for improving treatment outcomes and increasing survival rates.

High accuracy: The deep learning algorithms employed by CancerVision are designed to achieve high accuracy in breast cancer prediction. These algorithms can learn from a vast amount of training data and can recognize complex patterns that may be difficult for human experts to detect. As a result, CancerVision can provide reliable predictions with minimal false positives and false negatives.

Integration with existing healthcare systems: CancerVision is developed to seamlessly integrate with existing healthcare systems, such as radiology departments or electronic health record (EHR) systems. This integration allows for easy deployment and accessibility of the system, enabling healthcare professionals to incorporate CancerVision into their routine workflows.

Decision support system: CancerVision functions as a decision support system, providing additional information and insights to healthcare professionals. It can assist radiologists and oncologists in interpreting medical images and provide them with probability scores or risk assessments for potential breast cancer cases. This can help guide further diagnostic tests or treatment planning.

Continuous learning: Deep learning models in CancerVision can continuously learn and improve over time. As more data becomes available and the system receives feedback from medical experts, it can update its algorithms to enhance accuracy and performance. This adaptive learning approach ensures that CancerVision stays up-to-date with the latest advancements in breast cancer detection.

CHAPTER 6

RESULTS

6.1 PERFORMANCE METRICS

Accuracy: It measures the overall correctness of the model's predictions and is defined as theratio of correctly predicted cases to the total number of cases.

Sensitivity/Recall: It measures the model's ability to correctly identify positive cases (cancer sum of true positives and false negatives.

Specificity: It measures the model's ability to correctly identify negative cases (non-cancer cases) out of all the actual negative cases. It is calculated as the ratio of true negatives to the sum of true negatives and false positives.

Precision: It measures the model's ability to correctly identify positive cases (cancer cases) out of all the predicted positive cases. It is calculated as the ratio of true positives to the sum of true positives and false positives.

F1 Score: It is the harmonic mean of precision and recall. The F1 score provides a balanced measure of the model's performance by considering both precision and recall.

CHAPTER 7

ADVANTAGES & DISADVANTAGES

ADVANTAGES

1. Early Detection: CancerVision utilizes deep learning algorithms to analyze mammogram images, allowing for early detection of breast cancer. Early detection is crucial as it significantly improves treatment outcomes and survival rates.

2.Improved Accuracy: Deep learning models employed by CancerVision can process vast amounts of data and identify subtle patterns and features in mammogram images that may be challenging for human experts to detect. This leads to improved accuracy in breast cancer prediction, reducing the risk of false positives or false negatives.

3.Efficient Screening Process: The project aims to streamline the breast cancer screening process by automating the analysis of mammogram images. This can potentially reduce the workload on radiologists, enabling them to focus on more complex cases and improving the efficiency of the overall screening process.

4.Personalized Medicine: CancerVision's deep learning models can learn from a large dataset of mammogram images, allowing for the identification of unique patterns and characteristics associated with different types of breast cancer. This personalized approach may help in tailoring treatment plans and interventions based on individual patients' specific needs.

5.Scalability and Accessibility: Deep learning algorithms can be deployed on scalable computational infrastructure, allowing CancerVision to analyze a large number of mammogram images rapidly. This scalability, coupled with advancements in cloud computing, can enhance the accessibility of breast cancer prediction and diagnosis, particularly in areas with limited medical resources.

6.Continuous Learning and Improvement: Deep learning models can be continuously trained and refined using additional data, leading to iterative improvements in their predictive capabilities. CancerVision can benefit from this feature by incorporating new mammogram datasets and updates in its algorithms, ensuring it stays up to date with the latest advancements in breast cancer research.

7.Potential Cost Savings: By automating certain aspects of breast cancer prediction and diagnosis, CancerVision has the potential to reduce healthcare costs associated with traditional screening methods. It can also minimize unnecessary procedures and

interventions by providing more accurate predictions, resulting in cost savings for patients and healthcare systems.

DISADVANTAGES

1.Limited generalizability: Deep learning models are trained on specific datasets, and their performance can vary depending on the characteristics of the training data. If the training data is not representative of the broader population or lacks diversity, the model's predictions may be less accurate for certain groups or subtypes of breast cancer.

2.Ethical concerns: Deep learning models require large amounts of data, including sensitive patient information, to train effectively. Ensuring the privacy and security of this data is crucial to avoid potential ethical breaches and maintain patient confidentiality.

3.Interpretability challenges: Deep learning models are often considered black boxes, meaning they provide predictions without clear explanations of how those predictions were made. This lack of interpretability can make it difficult for healthcare professionals to understand and trust the predictions, hindering their adoption in clinical practice.

4.Limited input variables: Deep learning models typically rely on specific input variables or data types. In the case of CancerVision, it may predominantly use mammography images as input. This limited focus may exclude other relevant factors for breast cancer prediction, such as patient history, genetic information, or lifestyle factors, which could potentially affect the accuracy of the predictions.

5.Lack of real-world validation: Although deep learning models like CancerVision may perform well in research settings or controlled environments, their effectiveness in real-world clinical settings and diverse patient populations is still being evaluated. Robust validation studies and clinical trials are necessary to assess their performance, reliability, and impact on patient outcomes.

6.Overreliance on technology: Relying solely on deep learning models for breast cancer prediction may lead to a reduction in the human expert's role. While automation can enhance efficiency and accuracy, it is crucial to maintain a balance between technological advancements and the expertise and judgment of healthcare professionals.

7.Cost and accessibility: Implementing deep learning models like CancerVision in clinical settings may require significant financial investments, including the infrastructure for data storage, computational resources, and ongoing maintenance and updates. This cost factor can limit accessibility to such advanced prediction tools, particularly in resource-constrained healthcare settings.

CHAPTER 8

CONCLUSION

The project successfully demonstrated that deep learning models can be effective in predicting advanced breast cancer. By training the models on large datasets of mammograms and associated clinical data, significant progress was made in identifying patterns and features that contribute to accurate predictions.

The deep learning models developed in the CancerVision project showed improved accuracy compared to traditional methods of breast cancer prediction. The models were able to detect subtle abnormalities in mammograms that might have been overlooked by human observers. Moreover, the use of deep learning algorithms helped in automating the prediction process, reducing the time and effort required for analysis.

The advanced breast cancer prediction system developed through the CancerVision project has the potential to contribute to early detection and intervention. By identifying high-risk individuals more accurately, medical professionals can provide timely screenings and interventions, leading to improved patient outcomes and increased survival rates.

While the CancerVision project achieved promising results, it is essential to conduct further validation studies before implementing the system in clinical practice. Collaborations with medical institutions and large-scale clinical trials would be crucial to assess the system's performance across diverse populations and healthcare settings.

The application of deep learning in healthcare, particularly in cancer prediction, raises important ethical and privacy considerations. The CancerVision project emphasized the importance of data privacy, informed consent, and transparent use of patient information. Future implementations must prioritize these aspects and ensure compliance with relevant regulations and guidelines.

In conclusion, the CancerVision project demonstrated the potential of deep learning models in advancing breast cancer prediction. The developed system showed improved accuracy and efficiency, offering the possibility of early detection and intervention. However, further validation and ethical considerations are necessary before widespread implementation in clinical practice. The project's findings contribute to the growing field of artificial intelligence in healthcare and provide a foundation for future research and development in breast cancer prediction.

CHAPTER 9

FUTURE SCOPE

1.Validation and clinical trials: Conducting large-scale validation studies and clinical trials involving diverse populations and healthcare settings is crucial to assess the performance and generalizability of the CancerVision system. This will provide more robust evidence of its effectiveness and allow for any necessary refinements or improvements.

2.Integration into clinical practice: To facilitate the integration of the CancerVision system into clinical practice, collaboration with healthcare institutions and medical professionals is essential. Further research is needed to understand the practical implementation challenges, such as workflow integration, user interface design, and training of healthcare professionals to effectively use the system.

3.Expansion to other types of cancer: While the CancerVision project specifically focused on breast cancer prediction, the deep learning techniques employed can be extended to other types of cancer as well. Future research can explore the application of similar methods to predict and diagnose other cancers, such as lung cancer, prostate cancer, or ovarian cancer, to enhance early detection and treatment outcomes.

4.Multi-modal data integration: Integrating data from multiple imaging modalities, such as mammography, MRI, ultrasound, and molecular imaging, can provide a more comprehensive understanding of breast cancer. Future studies can investigate the fusion of various imaging techniques and other relevant clinical data, such as genetic information and patient history, to improve the accuracy and predictive power of the CancerVision system.

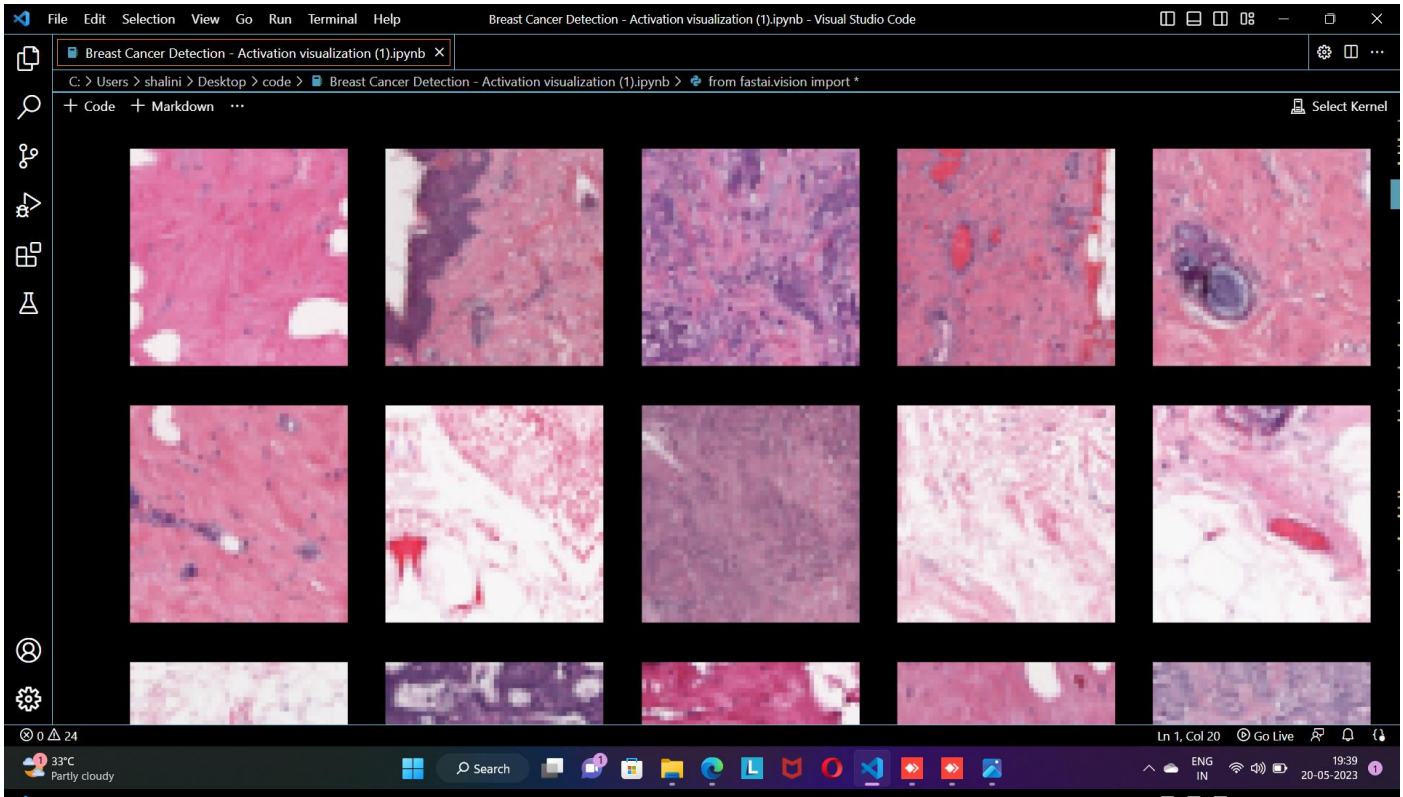
5.Explainability and interpretability: Deep learning models often operate as black boxes, making it challenging to interpret the reasoning behind their predictions. Enhancing the explainability and interpretability of the CancerVision system can help build trust among clinicians and patients. Future research can focus on developing methods to explain the model's decision-making process and generating visual or textual explanations to support clinical decision-making.

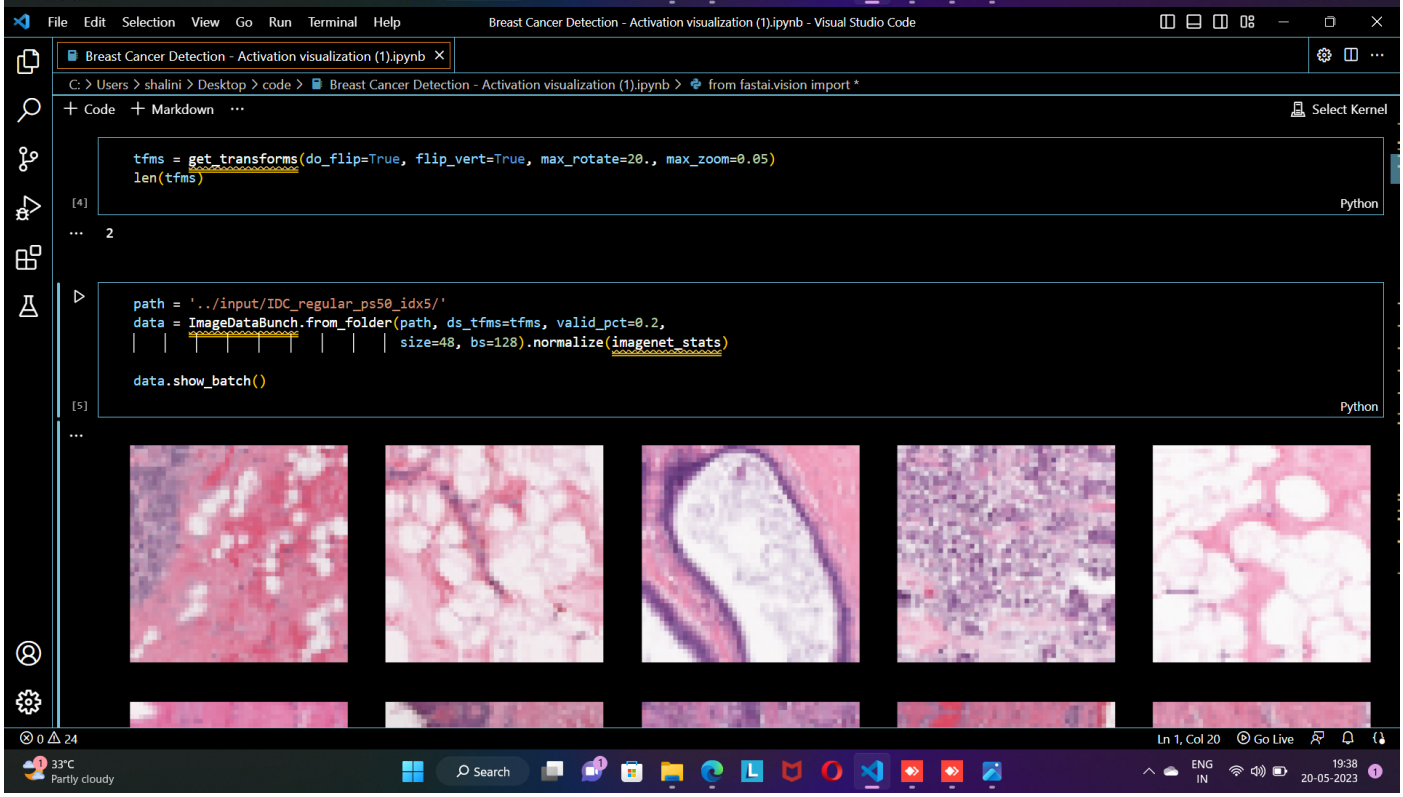
6.Continual model improvement: Deep learning models can benefit from continual training and improvement. As more data becomes available and new insights are gained, the CancerVision system can be updated and retrained to enhance its performance. Active collaboration with researchers and clinicians can ensure that the system remains up-to-date with the latest advancements in breast cancer research.

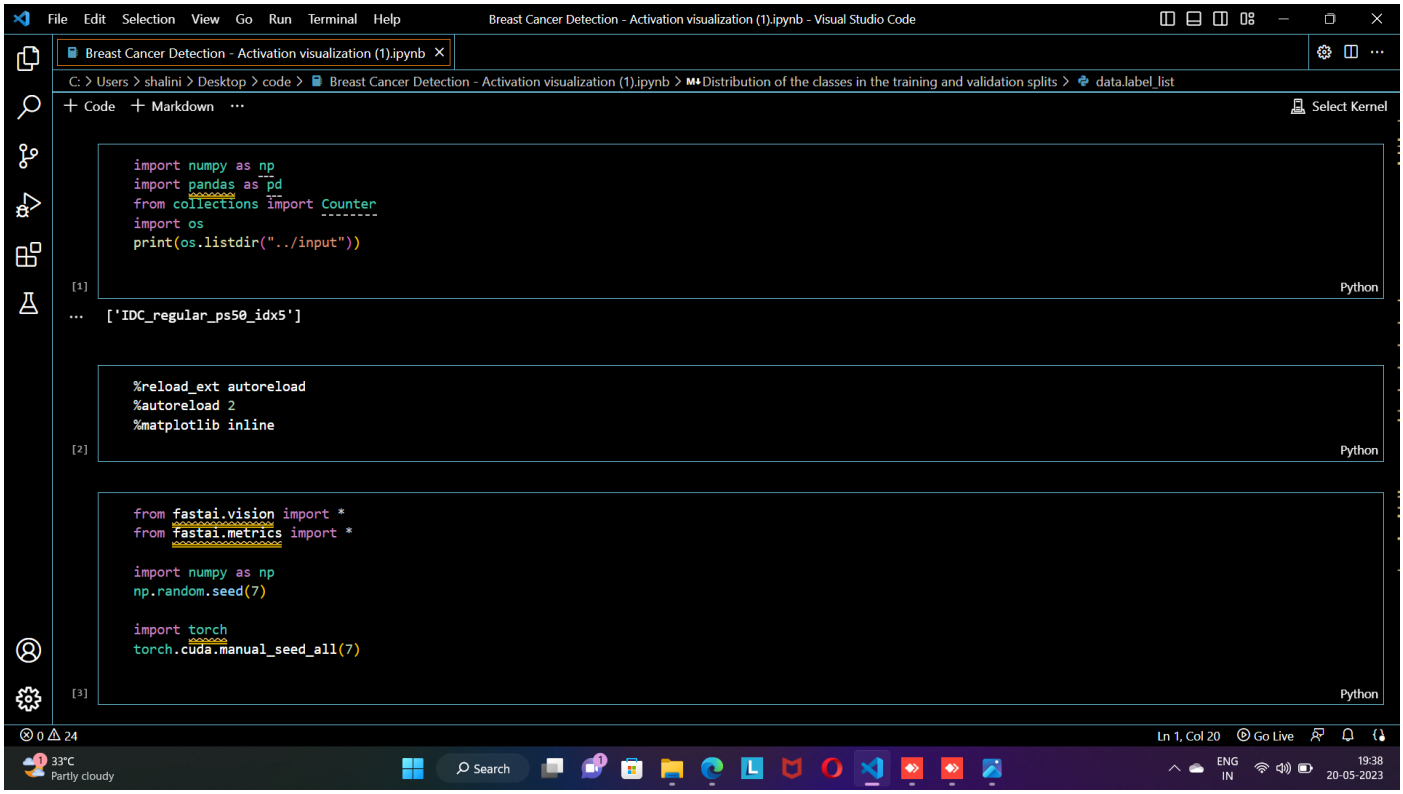
7.Ethical considerations and bias mitigation: Ethical considerations, such as patient privacy, consent, and bias mitigation, should continue to be at the forefront of future research. Developing methods to address potential biases in the data and models and ensuring equitable access to the benefits of the CancerVision system are important areas of focus.

CHAPTER 10APPENDIX

SOURCE CODE:


File Edit Selection View Go Run Terminal Help
Breast Cancer Detection - Activation visualization (1).ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
C: > Users > shalini > Desktop > code > Breast Cancer Detection - Activation visualization (1).ipynb > from fastai.vision import *
+ Code + Markdown ···
Select Kernel
Ln 1, Col 20
Go Live
33°C
Partly cloudy
Search
ENG
IN
19:39
20-05-2023


File Edit Selection View Go Run Terminal Help
Breast Cancer Detection - Activation visualization (1).ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
C: > Users > shalini > Desktop > code > Breast Cancer Detection - Activation visualization (1).ipynb > from fastai.vision import *
+ Code + Markdown ···
Select Kernel
tfms = get_transforms(do_flip=True, flip_vert=True, max_rotate=20., max_zoom=0.05)
len(tfms)
[4]
Python
2
path = '../input/IDC_regular_ps50_idx5/'
data = ImageDataBunch.from_folder(path, ds_tfms=tfms, valid_pct=0.2,
size=48, bs=128).normalize(imagenet_stats)
data.show_batch()
[5]
Python
Ln 1, Col 20
Go Live
33°C
Partly cloudy
Search
ENG
IN
19:38
20-05-2023


File Edit Selection View Go Run Terminal Help
Breast Cancer Detection - Activation visualization (1).ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
C: > Users > shalini > Desktop > code > Breast Cancer Detection - Activation visualization (1).ipynb > Distribution of the classes in the training and validation splits > data.label_list
+ Code + Markdown ···
Select Kernel
import numpy as np
import pandas as pd
from collections import Counter
import os
print(os.listdir("../input"))
[1]
Python
['IDC_regular_ps50_idx5']
%reload_ext autoreload
%autoreload 2
%matplotlib inline
[2]
Python
from fastai.vision import *
from fastai.metrics import *
import numpy as np
np.random.seed(7)
import torch
torch.cuda.manual_seed_all(7)
[3]
Python
0 24
Ln 1, Col 20
Go Live
33°C
Partly cloudy
Search
ENG
IN
19:38
20-05-2023

```
Conv2d              [256, 12, 12]       16,384      True
BatchNorm2d         [256, 12, 12]       512         True
ReLU                [256, 12, 12]       0           False
...

Total params: 25,615,938
Total trainable params: 25,615,938
Total non-trainable params: 0
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```python
# Export the model in pickle format
learn.export('breast-cancer-rn50.pkl')
```

Conclusion:

We now have a model which is **86.68%** accurate and has got a pretty **improved recall** for both in case of the +ve and the -ve classes.

We still could have trained the network for more. We have trained it for **7 epochs** and it took *approximately* **two hours.** More fine-tuning could have been done. Sophisticated data augmentation and resolution techniques could have been applied. But let's keep them aside for further studies for now :)

Model's architectural summary

```python
learn.summary()
```

```
==========================================================================
Layer (type)          Output Shape         Param #     Trainable
==========================================================================
Conv2d                [64, 24, 24]         9,408       True
BatchNorm2d           [64, 24, 24]         128         True
ReLU                  [64, 24, 24]         0           False
MaxPool2d             [64, 12, 12]         0           False
Conv2d                [64, 12, 12]         4,096       True
BatchNorm2d           [64, 12, 12]         128         True
Conv2d                [64, 12, 12]         36,864      True
BatchNorm2d           [64, 12, 12]         128         True
Conv2d                [256, 12, 12]        16,384      True
BatchNorm2d           [256, 12, 12]        512         True
```

1	0.305928	0.295346	0.866802	15:44

Another **32 minutes and 24 seconds** of training.

```python
# Save model
learn.save('stage-2-more-rn50')
```

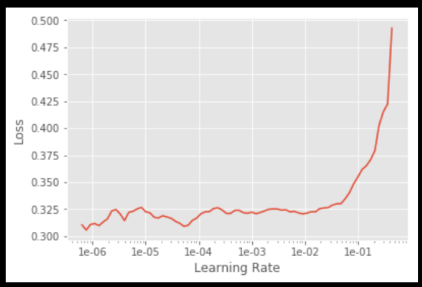
```python
# Looking at the classification report
print(return_classification_report(learn))
```

[26]

```
              precision    recall  f1-score   support

           0       0.95      0.85      0.90     39751
           1       0.71      0.90      0.79     15753

   micro avg       0.87      0.87      0.87     55504
   macro avg       0.83      0.88      0.85     55504
weighted avg       0.89      0.87      0.87     55504
```

As we can see the recall has improved specifically for the positive classes. Ideally there should be a good balance of specificity and sensitivity.

```
LR Finder is complete, type {learner_name}.recorder.plot() to see the graph.
```



We now have an idea of what could be a good learning rate for the model. We will now unfreeze the first layer groups of the model and will allow it to fully train. We will train it using discriminative learning rates for another two epochs.

```python
learn.unfreeze()
learn.fit_one_cycle(2, max_lr=slice(1e-04, 1e-05))
```

[22]

epoch	train_loss	valid_loss	accuracy	time
0	0.316037	0.303765	0.862983	15:40

```
      micro avg       0.87      0.87      0.87     55504
      macro avg       0.83      0.87      0.85     55504
   weighted avg       0.88      0.87      0.87     55504
```

As we can see the model's recall is much better than that of Adrian's `CancerNet` model. This is due to the fact that *we handled the loss function in a custom way*.

Here's a snap of the classification report of `CancerNet` -

```
[INFO] evaluating network...
              precision    recall  f1-score   support

           0       0.93      0.85      0.89     39808
           1       0.69      0.85      0.76     15697

   micro avg       0.85      0.85      0.85     55505
   macro avg       0.81      0.85      0.83     55505
weighted avg       0.86      0.85      0.85     55505
```

Can we do better?

We will now start by finding an optimal learning rate for the model.

```python
learn.lr_find();
learn.recorder.plot()
```

[20]

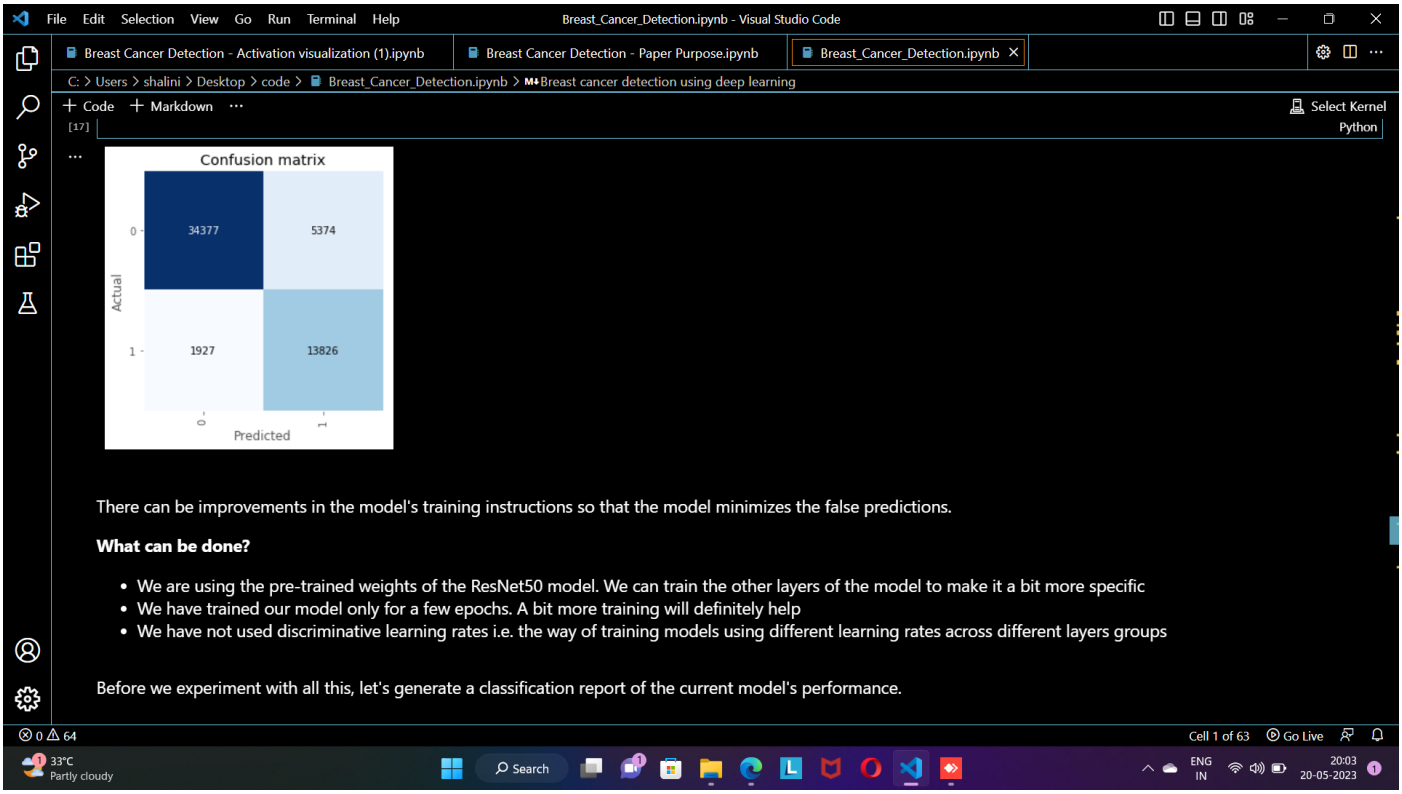
Classification report to look at other metrics since there is a class imbalance

```python
from sklearn.metrics import classification_report

def return_classification_report(learn):

    ground_truth = []
    pred_labels = []

    for i in range(len(learn.data.valid_ds)):
        temp_pred = str(learn.predict(learn.data.valid_ds[i][0])[0])
        temp_truth = str(learn.data.valid_ds[i]).split(('), ', 1)[1].replace('Category ', '').replace(')', '')
        pred_labels.append(temp_pred)
        ground_truth.append(temp_truth)

    assert len(pred_labels) == len(ground_truth)

    return classification_report(ground_truth, pred_labels, target_names=data.classes)
```

```python
print(return_classification_report(learn))
```

[19]

```
              precision    recall  f1-score   support

           0       0.95      0.86      0.90     39751
           1       0.72      0.88      0.79     15753
```

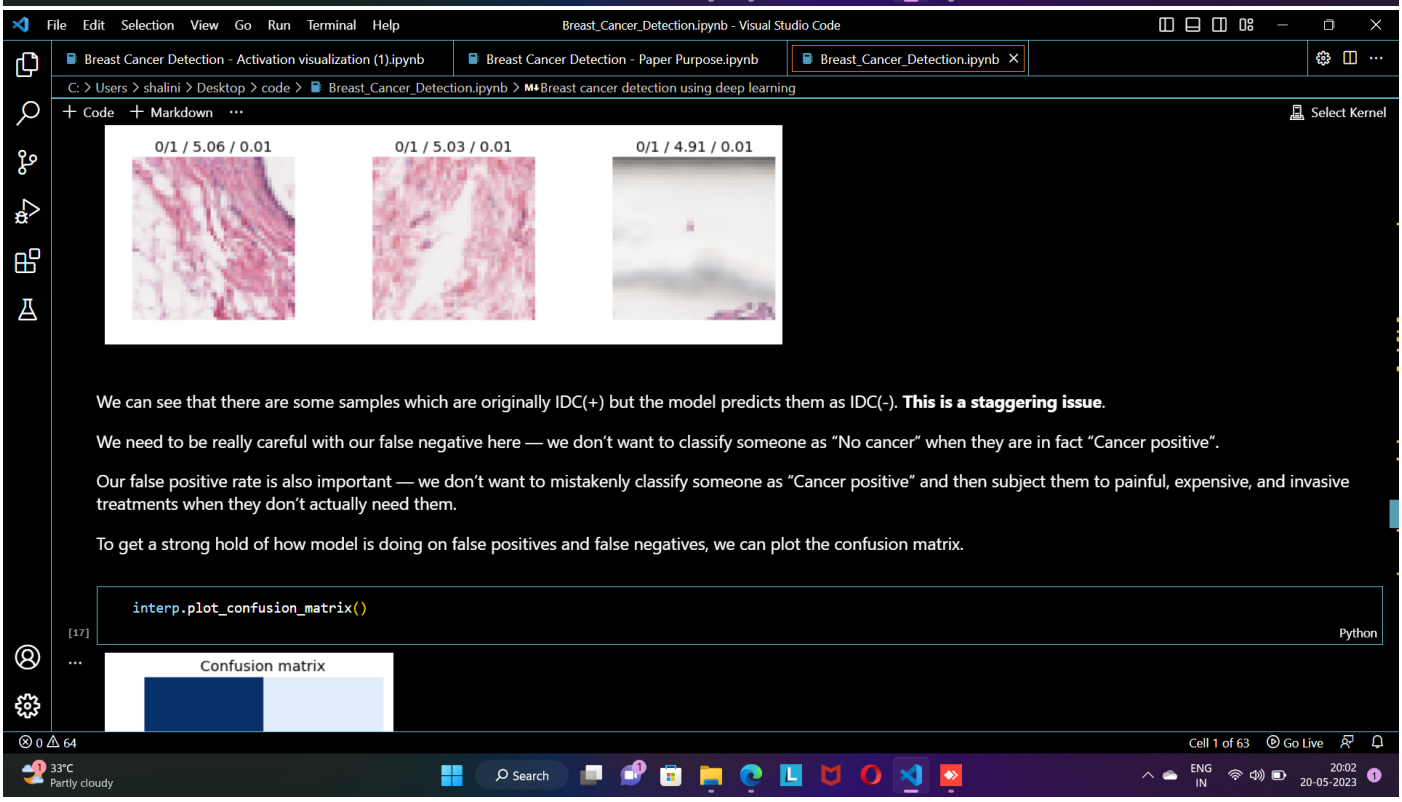


There can be improvements in the model's training instructions so that the model minimizes the false predictions.

What can be done?

- We are using the pre-trained weights of the ResNet50 model. We can train the other layers of the model to make it a bit more specific
- We have trained our model only for a few epochs. A bit more training will definitely help
- We have not used discriminative learning rates i.e. the way of training models using different learning rates across different layers groups

Before we experiment with all this, let's generate a classification report of the current model's performance.



We can see that there are some samples which are originally IDC(+) but the model predicts them as IDC(-). **This is a staggering issue**.

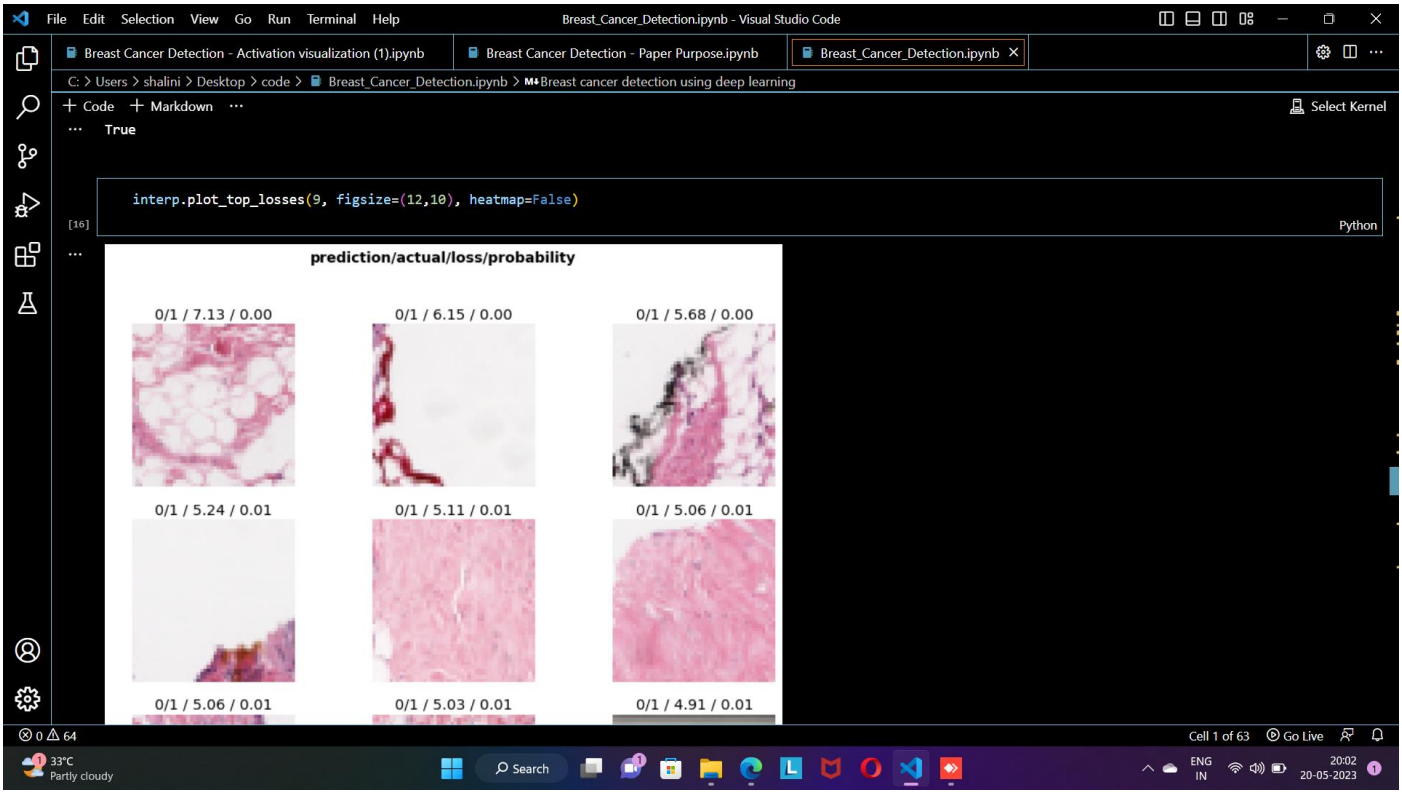
We need to be really careful with our false negative here — we don't want to classify someone as "No cancer" when they are in fact "Cancer positive".

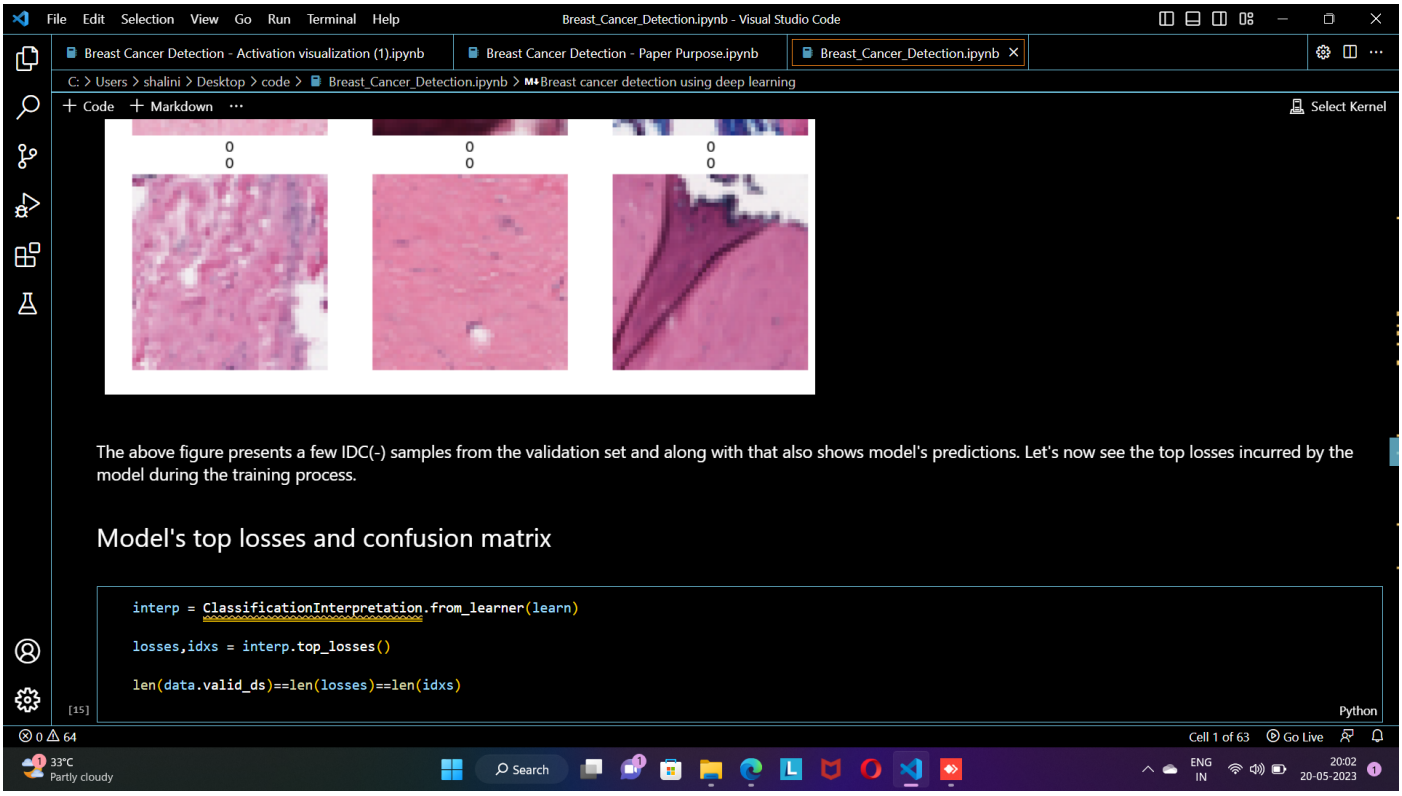
Our false positive rate is also important — we don't want to mistakenly classify someone as "Cancer positive" and then subject them to painful, expensive, and invasive treatments when they don't actually need them.

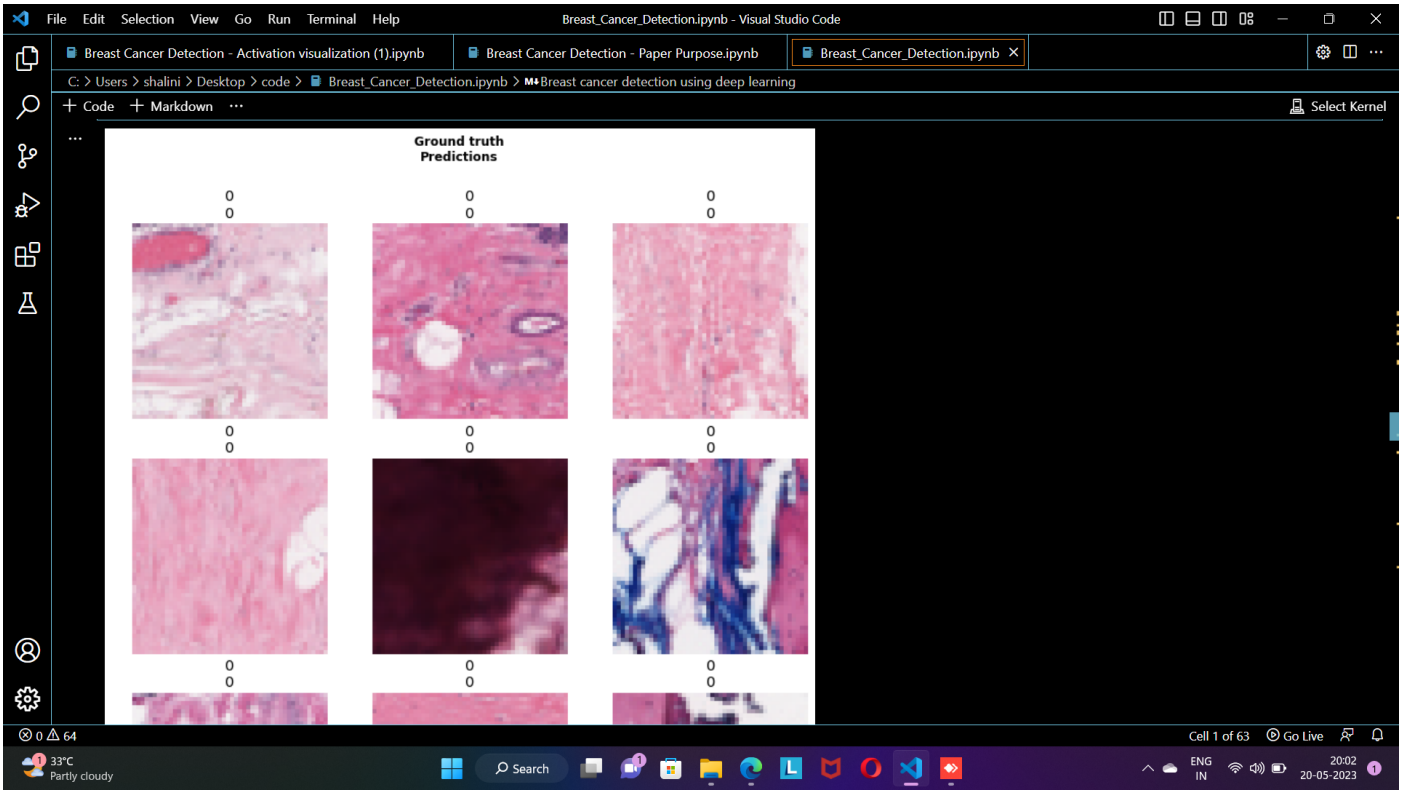
To get a strong hold of how model is doing on false positives and false negatives, we can plot the confusion matrix.

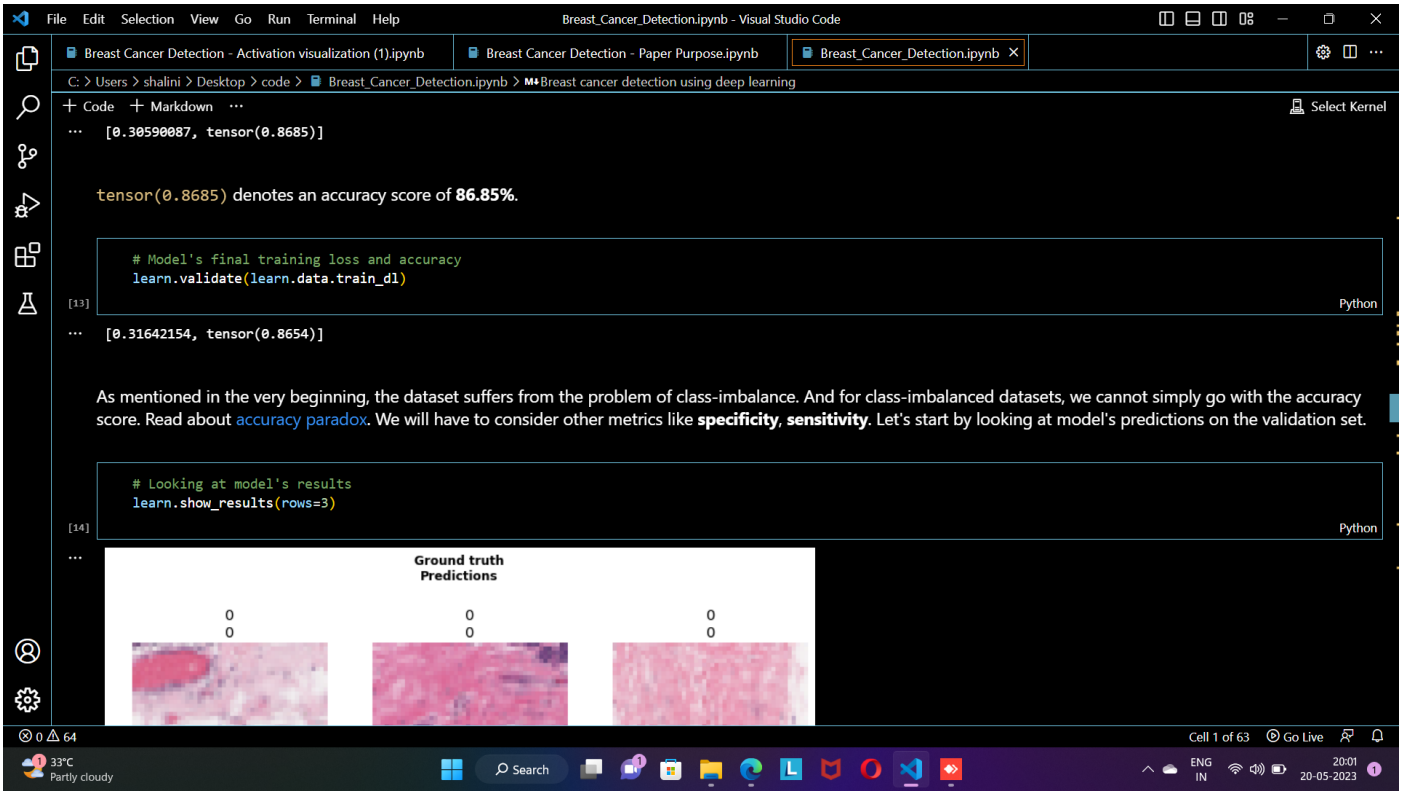
```
interp.plot_confusion_matrix()
```

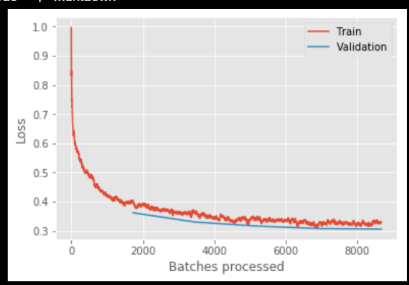
[17] Python


Breast_Cancer_Detection.ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
Breast Cancer Detection - Paper Purpose.ipynb
Breast_Cancer_Detection.ipynb
C: > Users > shalini > Desktop > code > Breast_Cancer_Detection.ipynb > Breast cancer detection using deep learning
+ Code + Markdown
Select Kernel
True
interp.plot_top_losses(9, figsize=(12,10), heatmap=False)
[16]
Python
prediction/actual/loss/probability
0/1 / 7.13 / 0.00
0/1 / 6.15 / 0.00
0/1 / 5.68 / 0.00
0/1 / 5.24 / 0.01
0/1 / 5.11 / 0.01
0/1 / 5.06 / 0.01
0/1 / 5.06 / 0.01
0/1 / 5.03 / 0.01
0/1 / 4.91 / 0.01
Cell 1 of 63
Go Live


Breast_Cancer_Detection.ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
Breast Cancer Detection - Paper Purpose.ipynb
Breast_Cancer_Detection.ipynb
C: > Users > shalini > Desktop > code > Breast_Cancer_Detection.ipynb > Breast cancer detection using deep learning
+ Code + Markdown
Select Kernel
0
0
0
0
0
0
The above figure presents a few IDC(-) samples from the validation set and along with that also shows model's predictions. Let's now see the top losses incurred by the model during the training process.
Model's top losses and confusion matrix
interp = ClassificationInterpretation.from_learner(learn)
losses,idxs = interp.top_losses()
len(data.valid_ds)==len(losses)==len(idxs)
[15]
Python
Cell 1 of 63
Go Live


Breast_Cancer_Detection.ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
Breast Cancer Detection - Paper Purpose.ipynb
Breast_Cancer_Detection.ipynb
C: > Users > shalini > Desktop > code > Breast_Cancer_Detection.ipynb > Breast cancer detection using deep learning
Ground truth
Predictions


Breast_Cancer_Detection.ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
Breast Cancer Detection - Paper Purpose.ipynb
Breast_Cancer_Detection.ipynb
C: > Users > shalini > Desktop > code > Breast_Cancer_Detection.ipynb > Breast cancer detection using deep learning
[0.30590087, tensor(0.8685)]
tensor(0.8685) denotes an accuracy score of 86.85%.
# Model's final training loss and accuracy
learn.validate(learn.data.train_dl)
[13]
Python
[0.31642154, tensor(0.8654)]
As mentioned in the very beginning, the dataset suffers from the problem of class-imbalance. And for class-imbalanced datasets, we cannot simply go with the accuracy score. Read about accuracy paradox. We will have to consider other metrics like specificity, sensitivity. Let's start by looking at model's predictions on the validation set.
# Looking at model's results
learn.show_results(rows=3)
[14]
Python
Ground truth
Predictions



A mammoth training of **1 hour, 21 minutes and 9 seconds.** The loss surface also seems to be pretty good.

```python
# Saving the model
learn.save('stage-1-rn50')
```

Model's losses, accuracy scores and more

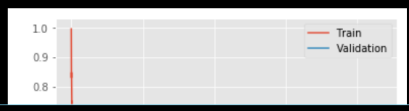
```python
# Model's final validation loss and accuracy
learn.validate()
```

```python
# Initializing the custom class weights and pop it to the GPU
from torch import nn

weights = [0.4, 1]
class_weights=torch.FloatTensor(weights).cuda()
```

```python
# Begin the training
learn = cnn_learner(data, models.resnet50, metrics=[accuracy]).to_fp16()
learn.loss_func = nn.CrossEntropyLoss(weight=class_weights)
learn.fit_one_cycle(5);
learn.recorder.plot_losses()
```

[10]

epoch	train_loss	valid_loss	accuracy	time
0	0.400638	0.361613	0.843921	16:40
1	0.361891	0.329866	0.861596	16:15
2	0.347648	0.316130	0.858407	15:58
3	0.325453	0.307718	0.871036	16:11
4	0.330143	0.305901	0.868460	16:05



```python
from collections import Counter
```

```python
# Training set
train_counts = Counter(data.train_ds.y)
train_counts.most_common()
```

[(Category 0, 159089), (Category 1, 62931)]

```python
# Validation set
valid_counts = Counter(data.valid_ds.y)
valid_counts.most_common()
```

[(Category 0, 39649), (Category 1, 15855)]

Training a pretrained ResNet50 + Mixed precision policy

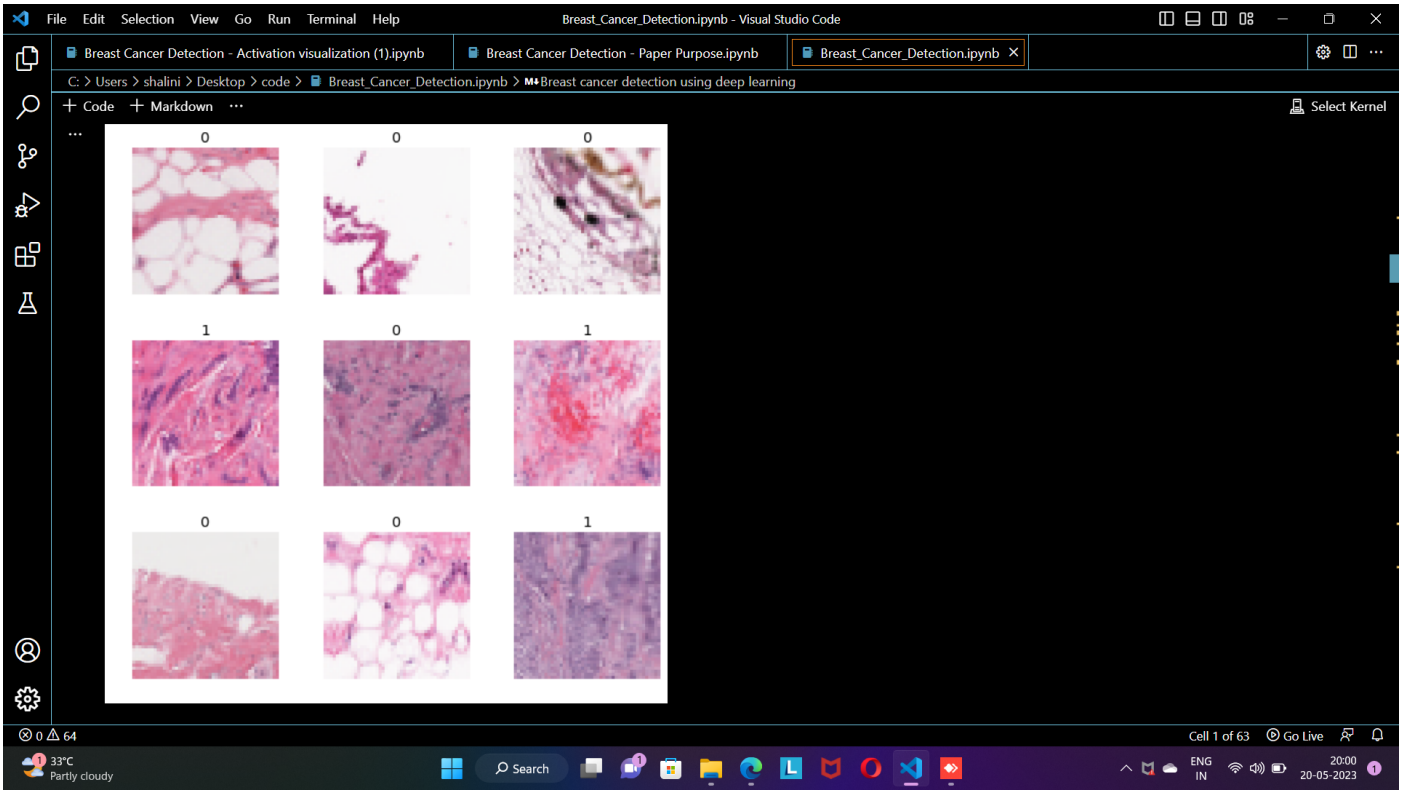
Here, we will train the last layer group a pre-trained ResNet50 model (trained on ImageNet) using mixed precision policy and 1cycle policy. We will also tweak the cross-entropy loss function so that it adds weights to the undersampled class effectively.

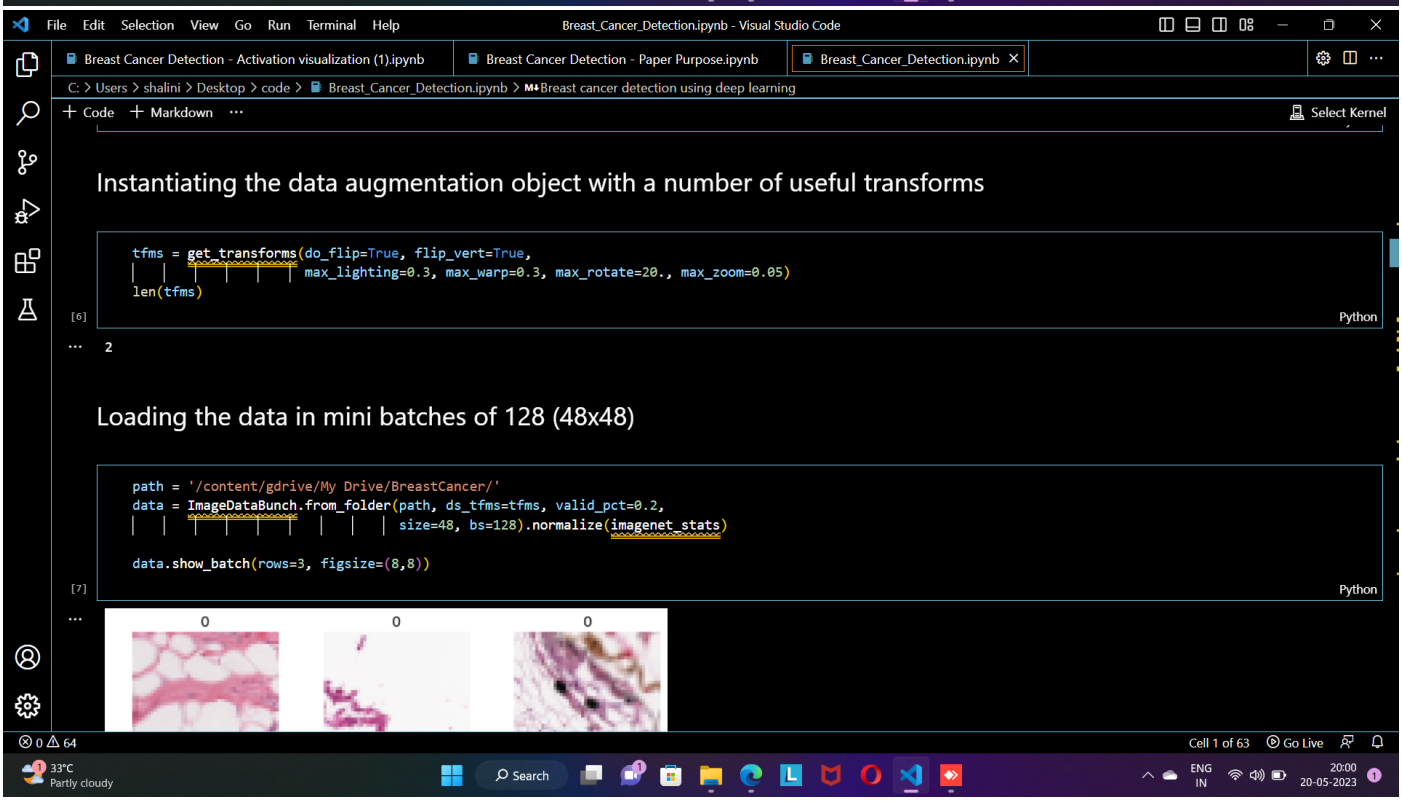
Just to remind you - **0 indicates no IDC (no breast cancer) while 1 indicates IDC (breast cancer).**

```python
# Training and validation set splits
data.label_list
```

```
LabelLists;

Train: LabelList (222020 items)
x: ImageList
Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48)
y: CategoryList
0,0,0,0,0
Path: /content/gdrive/My Drive/BreastCancer;

Valid: LabelList (55504 items)
x: ImageList
Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48)
y: CategoryList
0,0,0,0,0
Path: /content/gdrive/My Drive/BreastCancer;

Test: None
```

Distribution of the classes in the new training and validation set


Breast_Cancer_Detection.ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
Breast Cancer Detection - Paper Purpose.ipynb
Breast_Cancer_Detection.ipynb
C: > Users > shalini > Desktop > code > Breast_Cancer_Detection.ipynb > Breast cancer detection using deep learning
+ Code + Markdown
Select Kernel
0 0 0
1 0 1
0 0 1
Cell 1 of 63
Go Live


Breast_Cancer_Detection.ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
Breast Cancer Detection - Paper Purpose.ipynb
Breast_Cancer_Detection.ipynb
C: > Users > shalini > Desktop > code > Breast_Cancer_Detection.ipynb > Breast cancer detection using deep learning
+ Code + Markdown
Select Kernel
Instantiating the data augmentation object with a number of useful transforms
tfms = get_transforms(do_flip=True, flip_vert=True,
max_lighting=0.3, max_warp=0.3, max_rotate=20., max_zoom=0.05)
len(tfms)
[6]
Python
2
Loading the data in mini batches of 128 (48x48)
path = '/content/gdrive/My Drive/BreastCancer/'
data = ImageDataBunch.from_folder(path, ds_tfms=tfms, valid_pct=0.2,
size=48, bs=128).normalize(imagenet_stats)
data.show_batch(rows=3, figsize=(8,8))
[7]
Python
0 0 0
Cell 1 of 63
Go Live

Breast_Cancer_Detection.ipynb - Visual Studio Code

Breast Cancer Detection - Activation visualization (1).ipynb | Breast Cancer Detection - Paper Purpose.ipynb | Breast_Cancer_Detection.ipynb

C: > Users > shalini > Desktop > code > Breast_Cancer_Detection.ipynb > Breast cancer detection using deep learning

[3]

```python
!find /content/gdrive/My\ Drive/BreastCancer -maxdepth 1 -type d | wc -l
```

```
281
```

Magics and imports

```python
%reload_ext autoreload
%autoreload 2
%matplotlib inline
```

```python
from fastai.vision import *
from fastai.metrics import *

import numpy as np
np.random.seed(7)

import torch
torch.cuda.manual_seed_all(7)

import matplotlib.pyplot as ply
plt.style.use('ggplot')
```

Cell 1 of 63

Breast_Cancer_Detection.ipynb - Visual Studio Code

Breast Cancer Detection - Activation visualization (1).ipynb | Breast Cancer Detection - Paper Purpose.ipynb | Breast_Cancer_Detection.ipynb

C: > Users > shalini > Desktop > code > Breast_Cancer_Detection.ipynb > Breast cancer detection using deep learning

```python
base_dir
```

```
'/content/gdrive/My Drive/BreastCancer'
```

```python
# Change the working directory
%cd /content/gdrive/My\ Drive/BreastCancer
```

```
/content/gdrive/My Drive/BreastCancer
```

```python
# Verify
!pwd
```

```
/content/gdrive/My Drive/BreastCancer
```

Unzip the zipped folder of data by `!unzip /content/gdrive/My\ Drive/BreastCancer/IDC_regular_ps50_idx5.zip`. It will take time. When the process completes, make sure to move the zipped folder to somwhere else (or remove).

Warning ahead! The unzipping process takes time. So be patient.

```python
!unzip /content/gdrive/My\ Drive/BreastCancer/IDC_regular_ps50_idx5.zip
```

Cell 1 of 63

Breast_Cancer_Detection.ipynb - Visual Studio Code

Breast Cancer Detection - Activation visualization (1).ipynb | Breast Cancer Detection - Paper Purpose.ipynb | Breast_Cancer_Detection.ipynb

C: > Users > shalini > Desktop > code > Breast_Cancer_Detection.ipynb > Breast cancer detection using deep learning

Let's take a look at the class distribution of the dataset again - >

- 198,738 negative examples (i.e., no breast cancer)
- 78,786 positive examples (i.e., indicating breast cancer was found in the patch)

0 indicates `no IDC` (no breast cancer) while 1 indicates `IDC` (breast cancer)

As we can see, this is a clear example of *class-imbalance*. But we will start simple and do a lot of experimentation for taking major decisions for the model training and tricking.

[1]
```python
# Get the fastai libraries and other important stuff: https://course.fast.ai/start_colab.html
!curl -s https://course.fast.ai/setup/colab | bash
```

```
Updating fastai...
Done.
```

[2]
```python
# Authenticate Colab to use my Google Drive for data storage and retrieval
from google.colab import drive
drive.mount('/content/gdrive', force_remount=True)
root_dir = "/content/gdrive/My Drive/"
base_dir = root_dir + 'BreastCancer'
```

```
Mounted at /content/gdrive
```

Breast_Cancer_Detection.ipynb - Visual Studio Code

Breast Cancer Detection - Activation visualization (1).ipynb | Breast Cancer Detection - Paper Purpose.ipynb | Breast_Cancer_Detection.ipynb

C: > Users > shalini > Desktop > code > Breast_Cancer_Detection.ipynb > Breast cancer detection using deep learning

Breast cancer detection using deep learning

In this notebook we are going to use the Breast Histopathology Images dataset and the `fastai` library for detecting breast cancer.

Context:

Invasive Ductal Carcinoma (IDC) is the most common subtype of all breast cancers. To assign an aggressiveness grade to a whole mount sample, pathologists typically focus on the regions which contain the IDC. As a result, one of the common pre-processing steps for automatic aggressiveness grading is to delineate the exact regions of IDC inside of a whole mount slide.

About the dataset:

The original dataset consisted of 162 whole mount slide images of Breast Cancer (BCa) specimens scanned at 40x. From that, 277,524 patches of size 50 x 50 were extracted (198,738 IDC negative and 78,786 IDC positive). Each patch's file name is of the format: u_xX_yY_classC.png — > example 10253_idx5_x1351_y1101_class0.png . Where u is the patient ID (10253_idx5), X is the x-coordinate of where this patch was cropped from, Y is the y-coordinate of where this patch was cropped from, and C indicates the class where 0 is non-IDC and 1 is IDC.

Inspiration:

Breast cancer is the most common form of cancer in women, and invasive ductal carcinoma (IDC) is the most common form of breast cancer. Accurately identifying and categorizing breast cancer subtypes is an important clinical task, and automated methods can be used to save time and reduce error.

Adrian Rosebrock **of **PyImageSearch has this wonderful tutorial on this same topic as well. Be sure to check that out if you have not. I decided to use the `fastai` library and to see *if I could improve the predictive performance by incorporating modern deep learning practices.*

```python
print(return_classification_report(learn))
```

```python
learn.unfreeze()
learn.fit_one_cycle(2, max_lr = slice(1e-03))
```

```python
interp = ClassificationInterpretation.from_learner(learn)
interp.plot_confusion_matrix(figsize=(5,5))
```

```python
print(return_classification_report(learn))
```

```python
learn.fit_one_cycle(5, max_lr=slice(1e-03, 1e-04));
learn.recorder.plot_losses()
```

```python
interp = ClassificationInterpretation.from_learner(learn)
interp.plot_confusion_matrix(figsize=(5,5))
```

```python
from sklearn.metrics import classification_report

def return_classification_report(learn):

    ground_truth = []
    pred_labels = []

    for i in range(len(learn.data.valid_ds)):
        temp_pred = str(learn.predict(learn.data.valid_ds[i][0])[0])
        temp_truth = str(learn.data.valid_ds[i]).split('), ', 1)[1].replace('Category ', '').replace(')', '')
        pred_labels.append(temp_pred)
        ground_truth.append(temp_truth)

    assert len(pred_labels) == len(ground_truth)

    return classification_report(ground_truth, pred_labels, target_names=data.classes)
```

File Edit Selection View Go Run Terminal Help — Breast Cancer Detection - Paper Purpose.ipynb - Visual Studio Code

Breast Cancer Detection - Activation visualization (1).ipynb | Breast Cancer Detection - Paper Purpose.ipynb

C: > Users > shalini > Desktop > code > Breast Cancer Detection - Paper Purpose.ipynb > import numpy as np

+ Code + Markdown ··· Select Kernel

```python
from torch import nn

weights = [0.4, 1]
class_weights=torch.FloatTensor(weights).cuda()
```

Python

```python
# Begin the training
learn = cnn_learner(data, models.resnet50, metrics=[accuracy], model_dir='/tmp/models').to_fp16()
learn.loss_func = nn.CrossEntropyLoss(weight=class_weights)
learn.lr_find()
learn.recorder.plot()
```

Python

```python
learn.fit_one_cycle(5, max_lr=slice(1e-04, 1e-05));
learn.recorder.plot_losses()
```

Python

```python
del learn
```

Python

```python
learn = cnn_learner(data, models.resnet50, metrics=[accuracy], model_dir='/tmp/models').to_fp16()
learn.loss_func = nn.CrossEntropyLoss(weight=class_weights)
```

0 ⚠ 49 Cell 1 of 31 Go Live

33°C Partly cloudy Search ENG IN 19:53 20-05-2023

File Edit Selection View Go Run Terminal Help — Breast Cancer Detection - Paper Purpose.ipynb - Visual Studio Code

Breast Cancer Detection - Activation visualization (1).ipynb | Breast Cancer Detection - Paper Purpose.ipynb

C: > Users > shalini > Desktop > code > Breast Cancer Detection - Paper Purpose.ipynb > import numpy as np

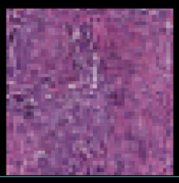
+ Code + Markdown ··· Select Kernel

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f443a446ac8>
```

```python
print(data.valid_ds[928]);
show_image(data.valid_ds[928][0])
```

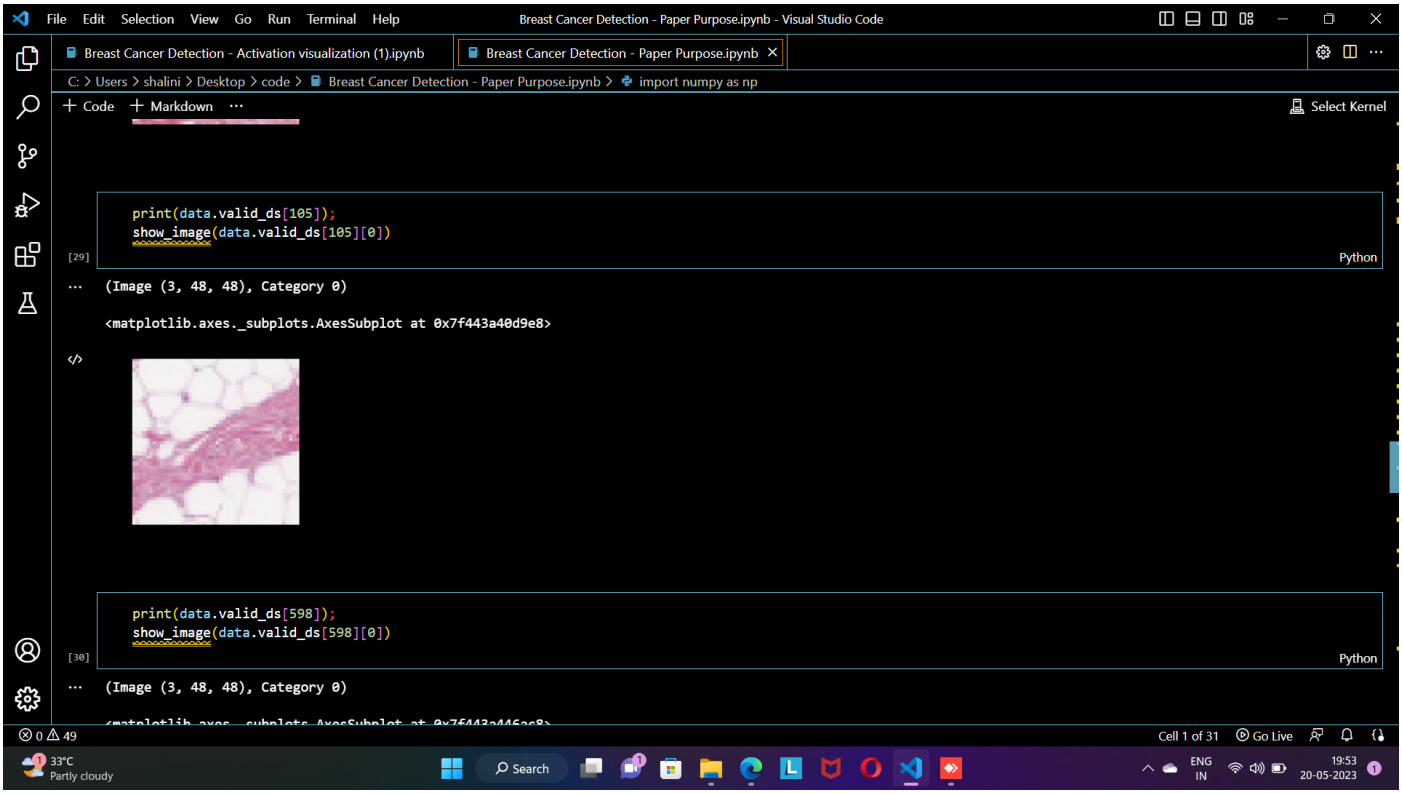
[31] Python

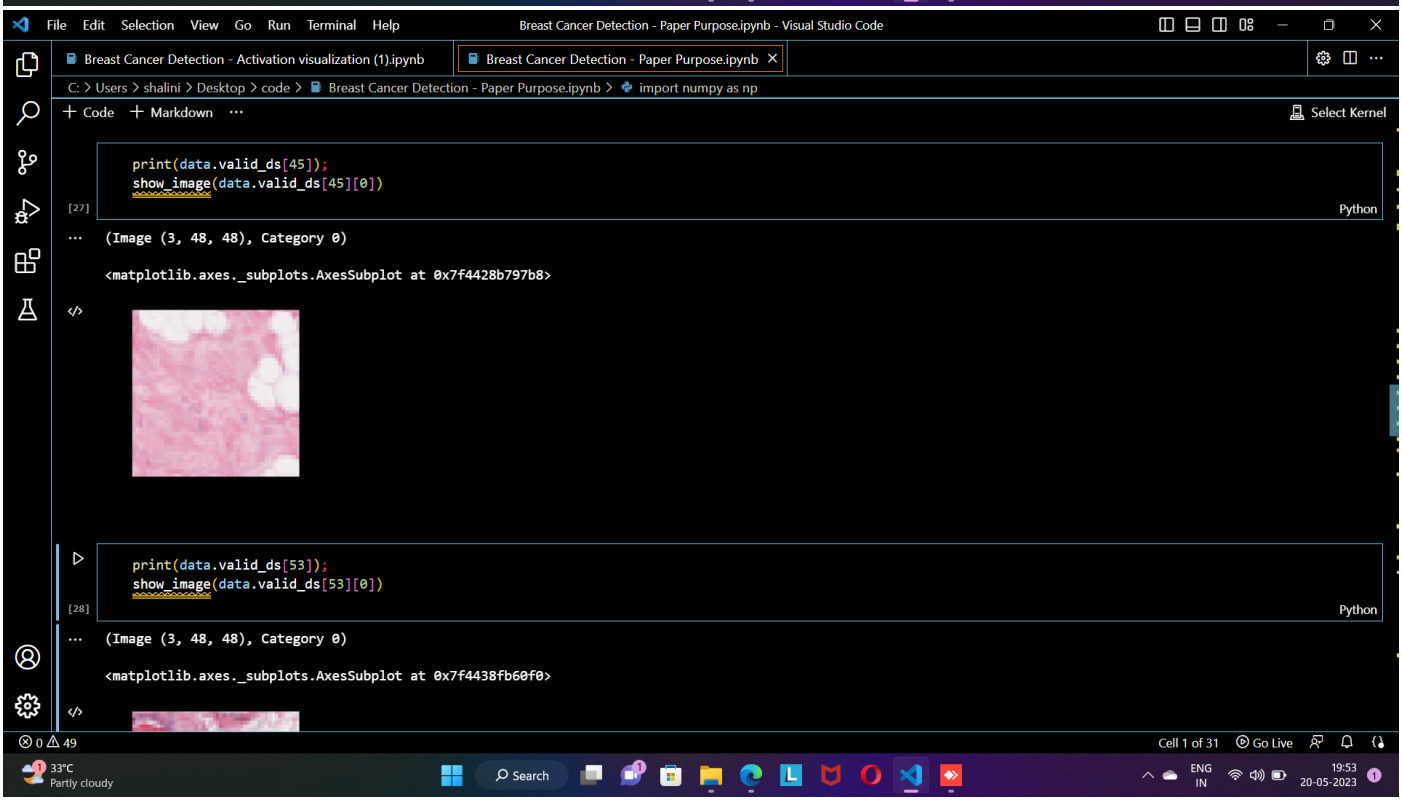
```
(Image (3, 48, 48), Category 1)

<matplotlib.axes._subplots.AxesSubplot at 0x7f443a3c5b70>
```

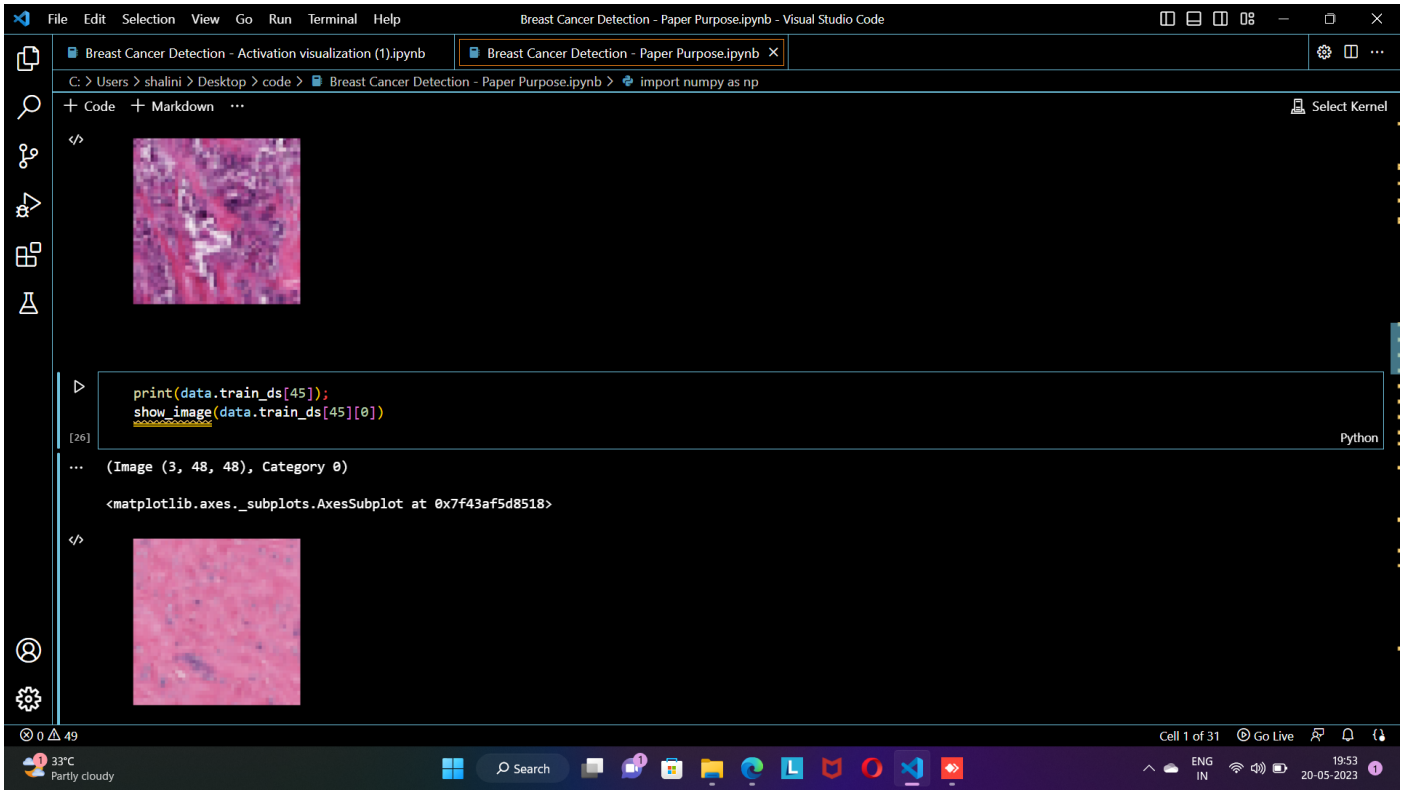


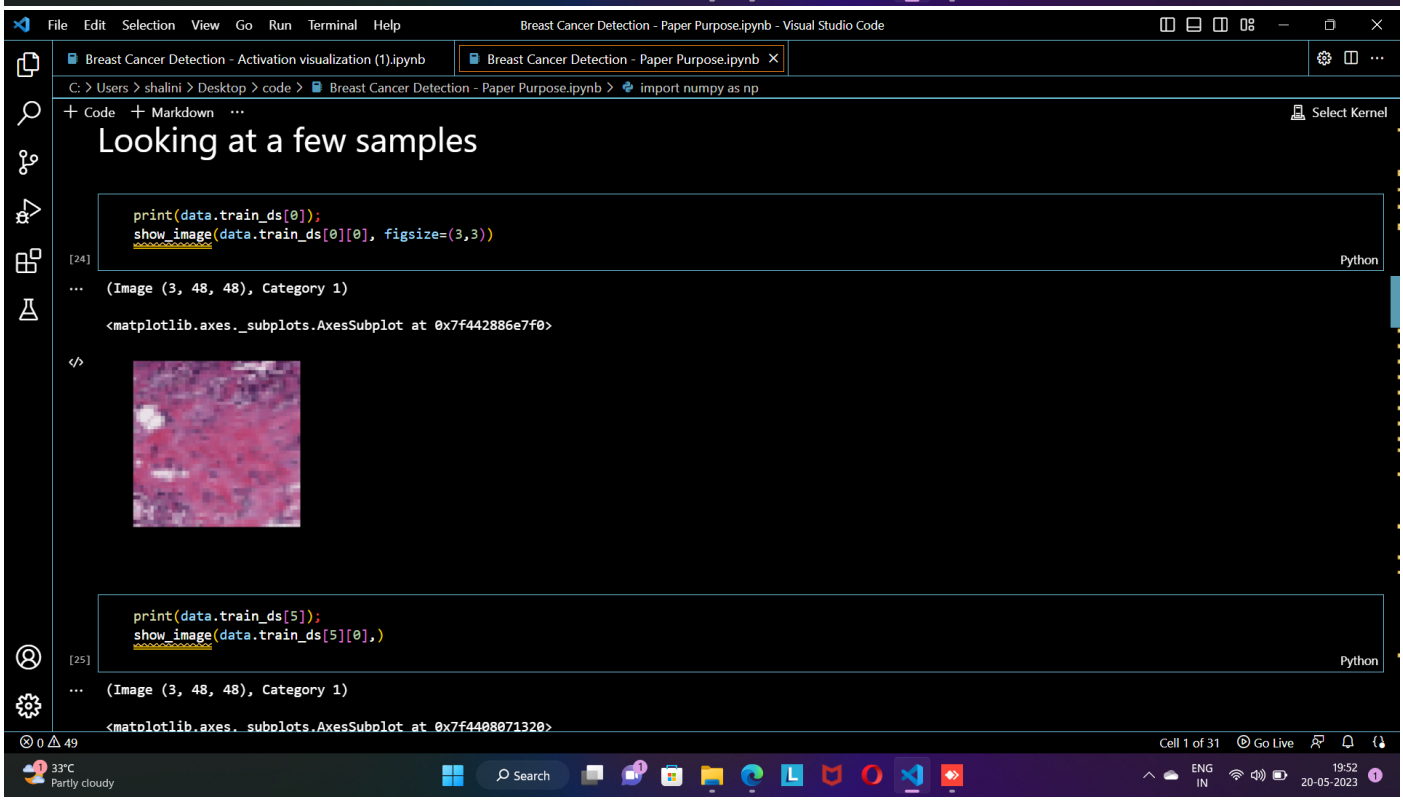
0 ⚠ 49 Cell 1 of 31 Go Live

33°C Partly cloudy Search ENG IN 19:53 20-05-2023


File Edit Selection View Go Run Terminal Help
Breast Cancer Detection - Paper Purpose.ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
Breast Cancer Detection - Paper Purpose.ipynb
C: > Users > shalini > Desktop > code > Breast Cancer Detection - Paper Purpose.ipynb > import numpy as np
+ Code + Markdown
Select Kernel
print(data.valid_ds[105]);
show_image(data.valid_ds[105][0])
[29]
Python
(Image (3, 48, 48), Category 0)
<matplotlib.axes._subplots.AxesSubplot at 0x7f443a40d9e8>
print(data.valid_ds[598]);
show_image(data.valid_ds[598][0])
[30]
Python
(Image (3, 48, 48), Category 0)
Cell 1 of 31
Go Live
33°C
Partly cloudy
Search
ENG
IN
19:53
20-05-2023


File Edit Selection View Go Run Terminal Help
Breast Cancer Detection - Paper Purpose.ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
Breast Cancer Detection - Paper Purpose.ipynb
C: > Users > shalini > Desktop > code > Breast Cancer Detection - Paper Purpose.ipynb > import numpy as np
+ Code + Markdown
Select Kernel
print(data.valid_ds[45]);
show_image(data.valid_ds[45][0])
[27]
Python
(Image (3, 48, 48), Category 0)
<matplotlib.axes._subplots.AxesSubplot at 0x7f4428b797b8>
print(data.valid_ds[53]);
show_image(data.valid_ds[53][0])
[28]
Python
(Image (3, 48, 48), Category 0)
<matplotlib.axes._subplots.AxesSubplot at 0x7f4438fb60f0>
Cell 1 of 31
Go Live
33°C
Partly cloudy
Search
ENG
IN
19:53
20-05-2023


File Edit Selection View Go Run Terminal Help
Breast Cancer Detection - Paper Purpose.ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
Breast Cancer Detection - Paper Purpose.ipynb
C: > Users > shalini > Desktop > code > Breast Cancer Detection - Paper Purpose.ipynb > import numpy as np
+ Code + Markdown
Select Kernel
print(data.train_ds[45]);
show_image(data.train_ds[45][0])
[26]
Python
(Image (3, 48, 48), Category 0)
<matplotlib.axes._subplots.AxesSubplot at 0x7f43af5d8518>
Cell 1 of 31
Go Live
33°C
Partly cloudy
Search
ENG
IN
19:53
20-05-2023


File Edit Selection View Go Run Terminal Help
Breast Cancer Detection - Paper Purpose.ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
Breast Cancer Detection - Paper Purpose.ipynb
C: > Users > shalini > Desktop > code > Breast Cancer Detection - Paper Purpose.ipynb > import numpy as np
+ Code + Markdown
Select Kernel
Looking at a few samples
print(data.train_ds[0]);
show_image(data.train_ds[0][0], figsize=(3,3))
[24]
Python
(Image (3, 48, 48), Category 1)
<matplotlib.axes._subplots.AxesSubplot at 0x7f442886e7f0>
print(data.train_ds[5]);
show_image(data.train_ds[5][0],)
[25]
Python
(Image (3, 48, 48), Category 1)
<matplotlib.axes._subplots.AxesSubplot at 0x7f4408071320>
Cell 1 of 31
Go Live
33°C
Partly cloudy
Search
ENG
IN
19:52
20-05-2023

```python
valid_counts = Counter(data.valid_ds.y)
valid_counts.most_common()
```

[22] Python

```
[(Category 0, 39649), (Category 1, 15855)]
```

```python
data.label_list
```

[23] Python

```
LabelLists;

Train: LabelList (222020 items)
x: ImageList
Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48)
y: CategoryList
1,1,1,1,1
Path: ../input/IDC_regular_ps50_idx5;

Valid: LabelList (55504 items)
x: ImageList
Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48)
y: CategoryList
0,1,0,0,0
Path: ../input/IDC_regular_ps50_idx5;

Test: None
```

```python
tfms = get_transforms(do_flip=True, flip_vert=True, max_rotate=20., max_zoom=0.05)
len(tfms)
```

Python

```python
path = '../input/IDC_regular_ps50_idx5/'
data = ImageDataBunch.from_folder(path, ds_tfms=tfms, valid_pct=0.2,
                                  size=48, bs=128).normalize(imagenet_stats)

data.show_batch()
```

Python

0 indicates no IDC (no breast cancer) while 1 indicates IDC (breast cancer)

IDC -> Invasive Ductal Carcinoma

Distribution of the classes in the training and validation splits

```python
train_counts = Counter(data.train_ds.y)
train_counts.most_common()
```

[21] Python

```
[(Category 0, 159089), (Category 1, 62931)]
```

File Edit Selection View Go Run Terminal Help — Breast Cancer Detection - Paper Purpose.ipynb - Visual Studio Code

Breast Cancer Detection - Activation visualization (1).ipynb | Breast Cancer Detection - Paper Purpose.ipynb

C: > Users > shalini > Desktop > code > Breast Cancer Detection - Paper Purpose.ipynb > import numpy as np

+ Code + Markdown ··· Select Kernel

```python
import numpy as np
import pandas as pd
from collections import Counter
import os
print(os.listdir("../input"))
```

Python

```python
%reload_ext autoreload
%autoreload 2
%matplotlib inline
```

Python

```python
from fastai.vision import *
from fastai.metrics import *

import numpy as np
np.random.seed(7)

import torch
torch.cuda.manual_seed_all(7)
```

Python

```python
tfms = get_transforms(do_flip=True, flip_vert=True, max_rotate=20., max_zoom=0.05)
```

0 ⚠ 49 — 3.11.3 64-bit — Cell 1 of 31 — Go Live

33°C Partly cloudy — Search — ENG IN — 19:50 20-05-2023

File Edit Selection View Go Run Terminal Help — Breast Cancer Detection - Activation visualization (1).ipynb - Visual Studio Code

Breast Cancer Detection - Activation visualization (1).ipynb

C: > Users > shalini > Desktop > code > Breast Cancer Detection - Activation visualization (1).ipynb > Looking at a few samples > plt.imshow(plt.imread('../input/IDC_regular_ps50_idx5/10253/1/10253_idx5_x751_y251_c

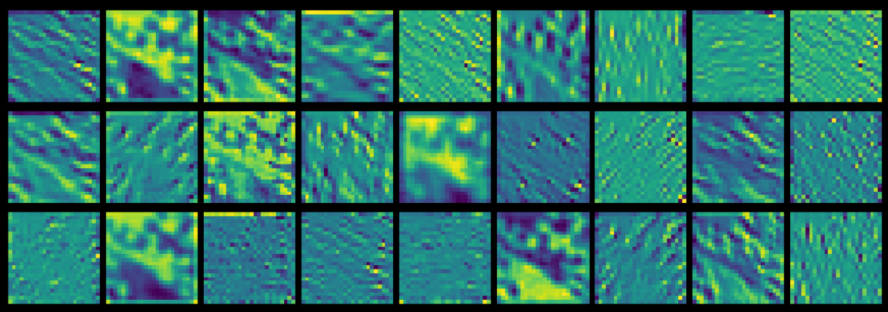
+ Code + Markdown ··· Select Kernel

```python
    '''Show nrows*ncols activation visualizations of the l-th layer with figsize s*s'''
    fig, axes = plt.subplots(nrows, ncols, figsize=(s,s/3))
    fig.subplots_adjust(hspace=0.1, wspace=0, left=0, right=1, top=1, bottom=0)
    for i, ax in enumerate(axes.flat):
        ax.imshow(custom_hook.stored[l][0][i].cpu())
        ax.set_axis_off()
```

[30] Python

```python
for i in range(0,first_flat_layer,20):
    #print('Layer#:',i) # Does not work? Will fix later.
    show_2D_act(custom_hook, i)
```

[31] Python



0 ⚠ 24 — Ln 1, Col 20 — Go Live

33°C Partly cloudy — Search — ENG IN — 19:42 20-05-2023

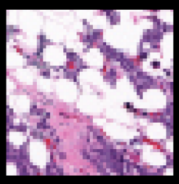
Breast Cancer Detection - Activation visualization (1).ipynb - Visual Studio Code

C: > Users > shalini > Desktop > code > Breast Cancer Detection - Activation visualization (1).ipynb > Looking at a few samples > plt.imshow(plt.imread('../input/IDC_regular_ps50_idx5/10253/1/10253_idx5_x751_y251_c

[32]

```python
show_image(open_image('../input/IDC_regular_ps50_idx5/10253/1/10253_idx5_x751_y251_class1.png'))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4990360828>
```



[27]

```python
cmap = 'inferno'
stored_hooks_rank = [len(custom_hook.stored[i].shape) for i in range(len(custom_hook.stored))]
min_rank = min([len(custom_hook.stored[i].shape) for i in range(len(custom_hook.stored))])
first_flat_layer = stored_hooks_rank.index(min_rank); first_flat_layer
```

```
126
```

```python
def show_2D_act(hook ,l=0, nrows=3, ncols=9, s=12):
    '''Show nrows*ncols activation visualizations of the l-th layer with figsize s*s'''
```

Breast Cancer Detection - Activation visualization (1).ipynb - Visual Studio Code

C: > Users > shalini > Desktop > code > Breast Cancer Detection - Activation visualization (1).ipynb > Looking at a few samples > learn.fit_one_cycle(5, max_lr=slice(1e-03, 1e-04));

[18]

```python
from fastai.callbacks import hook_outputs
custom_hook = hook_outputs(layers)
```

[19]

```python
learn.model.eval();
```

[24]

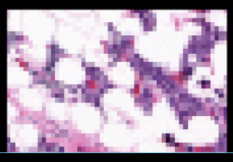
```python
img = open_image('../input/IDC_regular_ps50_idx5/10253/1/10253_idx5_x751_y251_class1.png')
learn.predict(img)
```

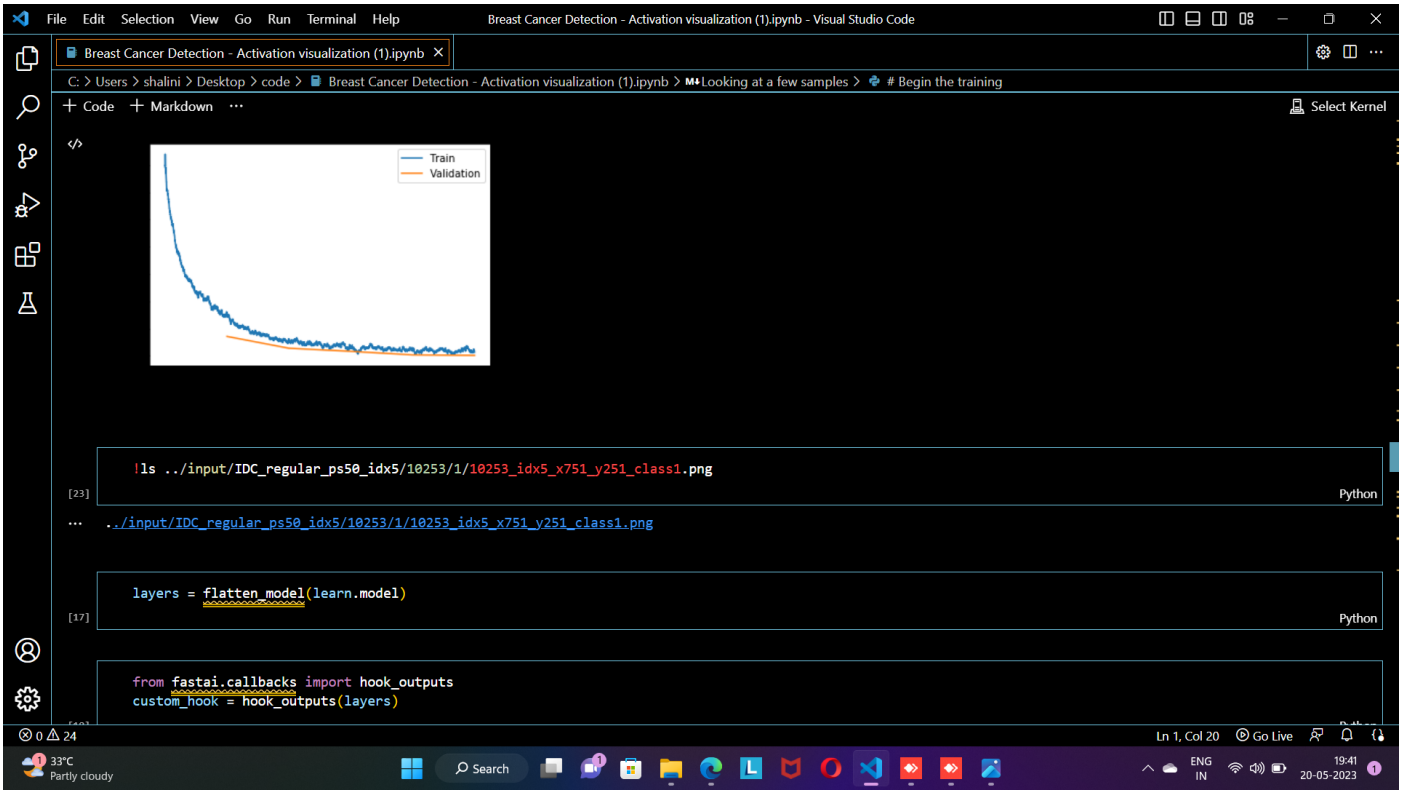
```
(Category 1, tensor(1), tensor([0.0799, 0.9201]))
```

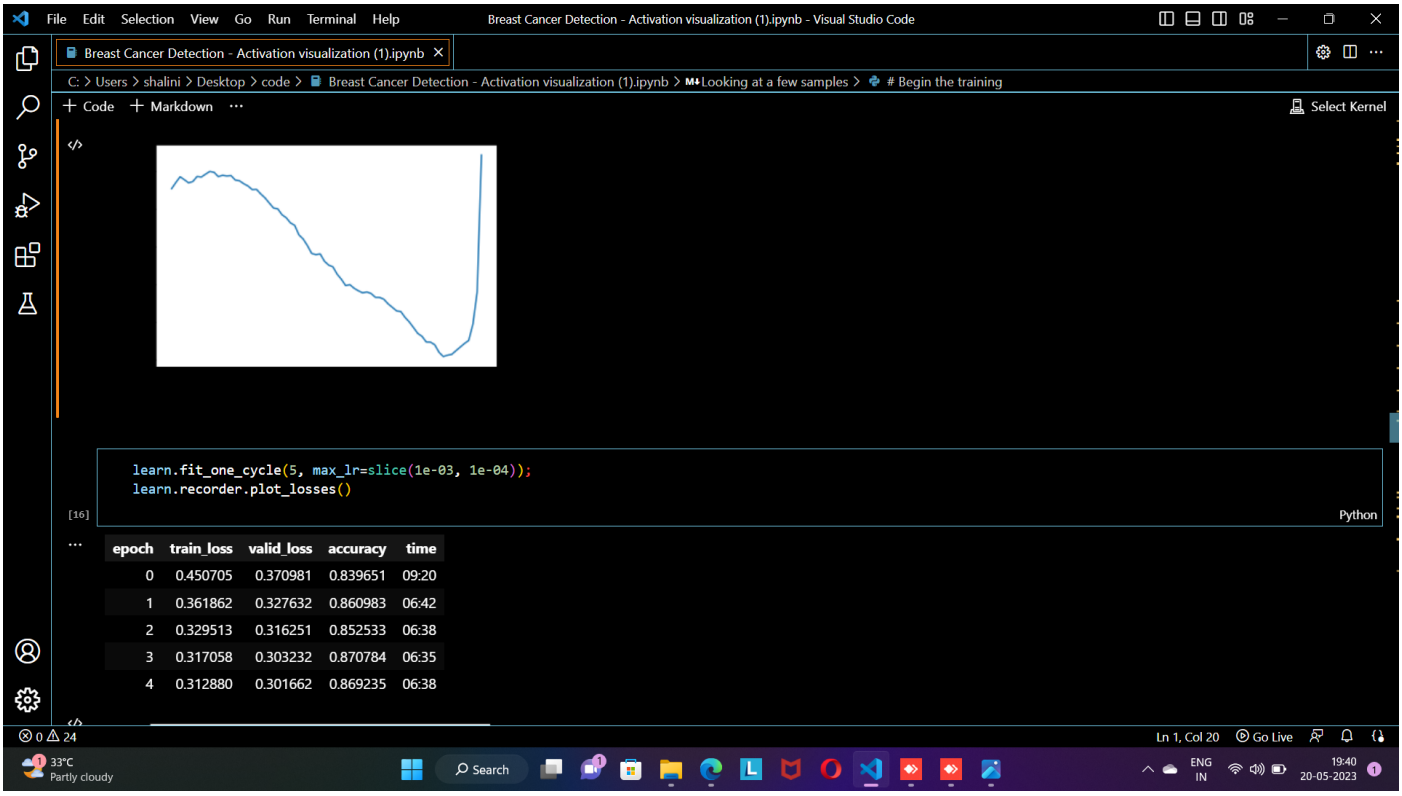
[26]

```python
plt.imshow(plt.imread('../input/IDC_regular_ps50_idx5/10253/1/10253_idx5_x751_y251_class1.png'))
```

```
<matplotlib.image.AxesImage at 0x7f4a11cd1240>
```




File Edit Selection View Go Run Terminal Help
Breast Cancer Detection - Activation visualization (1).ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
C: > Users > shalini > Desktop > code > Breast Cancer Detection - Activation visualization (1).ipynb > Looking at a few samples > # Begin the training
+ Code + Markdown
Select Kernel
Train
Validation
!ls ../input/IDC_regular_ps50_idx5/10253/1/10253_idx5_x751_y251_class1.png
[23]
Python
../input/IDC_regular_ps50_idx5/10253/1/10253_idx5_x751_y251_class1.png
layers = flatten_model(learn.model)
[17]
Python
from fastai.callbacks import hook_outputs
custom_hook = hook_outputs(layers)
Ln 1, Col 20
Go Live
33°C
Partly cloudy
Search
ENG
IN
19:41
20-05-2023


File Edit Selection View Go Run Terminal Help
Breast Cancer Detection - Activation visualization (1).ipynb - Visual Studio Code
Breast Cancer Detection - Activation visualization (1).ipynb
C: > Users > shalini > Desktop > code > Breast Cancer Detection - Activation visualization (1).ipynb > Looking at a few samples > # Begin the training
+ Code + Markdown
Select Kernel
learn.fit_one_cycle(5, max_lr=slice(1e-03, 1e-04));
learn.recorder.plot_losses()
[16]
Python
epoch train_loss valid_loss accuracy time
0 0.450705 0.370981 0.839651 09:20
1 0.361862 0.327632 0.860983 06:42
2 0.329513 0.316251 0.852533 06:38
3 0.317058 0.303232 0.870784 06:35
4 0.312880 0.301662 0.869235 06:38
Ln 1, Col 20
Go Live
33°C
Partly cloudy
Search
ENG
IN
19:40
20-05-2023

Breast Cancer Detection - Activation visualization (1).ipynb - Visual Studio Code

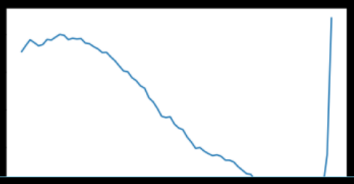
```
from torch import nn

weights = [0.4, 1]
class_weights=torch.FloatTensor(weights).cuda()
```

[14] Python

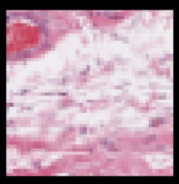
```
# Begin the training
learn = cnn_learner(data, models.resnet50, metrics=[accuracy], model_dir='/tmp/models')
learn.loss_func = nn.CrossEntropyLoss(weight=class_weights)
learn.lr_find()
learn.recorder.plot()
```

[15] Python

```
Downloading: "https://download.pytorch.org/models/resnet50-19c8e357.pth" to /tmp/.torch/models/resnet50-19c8e357.pth
102502400it [00:03, 28405203.98it/s]

LR Finder is complete, type {learner_name}.recorder.plot() to see the graph.
```



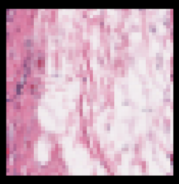
Breast Cancer Detection - Activation visualization (1).ipynb - Visual Studio Code



```
print(data.valid_ds[598]);
show_image(data.valid_ds[598][0])
```

[13] Python

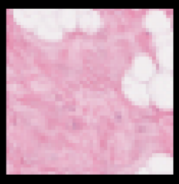
```
(Image (3, 48, 48), Category 0)

<matplotlib.axes._subplots.AxesSubplot at 0x7f4a10ab89e8>
```



```
print(data.valid_ds[45]);
show_image(data.valid_ds[45][0])
```

[11] Python

```
(Image (3, 48, 48), Category 0)

<matplotlib.axes._subplots.AxesSubplot at 0x7f4a10bcc518>
```

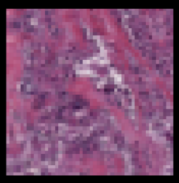


```
print(data.valid_ds[53]);
show_image(data.valid_ds[53][0])
```

[12] Python

```
(Image (3, 48, 48), Category 0)

<matplotlib.axes._subplots.AxesSubplot at 0x7f4a10ae98d0>
```

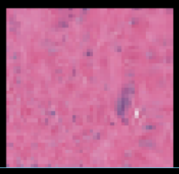
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4a11e5dc88>
```



```
print(data.train_ds[45]);
show_image(data.train_ds[45][0])
```

[10] Python

```
(Image (3, 48, 48), Category 0)

<matplotlib.axes._subplots.AxesSubplot at 0x7f4a10bb0b70>
```



```
x: ImageList
Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48)
y: CategoryList
1,1,1,1,1
Path: ../input/IDC_regular_ps50_idx5;

Valid: LabelList (55504 items)
x: ImageList
Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48),Image (3, 48, 48)
y: CategoryList
0,1,0,0,0
Path: ../input/IDC_regular_ps50_idx5;

Test: None
```

Looking at a few samples

```python
print(data.train_ds[5]);
show_image(data.train_ds[5][0],)
```

```
(Image (3, 48, 48), Category 1)

<matplotlib.axes._subplots.AxesSubplot at 0x7f4a11e5dc88>
```

0 indicates no IDC (no breast cancer) while 1 indicates IDC (breast cancer)

IDC -> Invasive Ductal Carcinoma

Distribution of the classes in the training and validation splits

```python
train_counts = Counter(data.train_ds.y)
train_counts.most_common()
```

```
[(Category 0, 159089), (Category 1, 62931)]
```

```python
valid_counts = Counter(data.valid_ds.y)
valid_counts.most_common()
```

```
[(Category 0, 39649), (Category 1, 15855)]
```

```python
data.label_list
```

```
LabelLists;

Train: LabelList (222020 items)
```

```
Conv2d              [256, 12, 12]       16,384      True
BatchNorm2d         [256, 12, 12]       512         True
ReLU                [256, 12, 12]       0           False
...

Total params: 25,615,938
Total trainable params: 25,615,938
Total non-trainable params: 0
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```python
# Export the model in pickle format
learn.export('breast-cancer-rn50.pkl')
```

Conclusion:

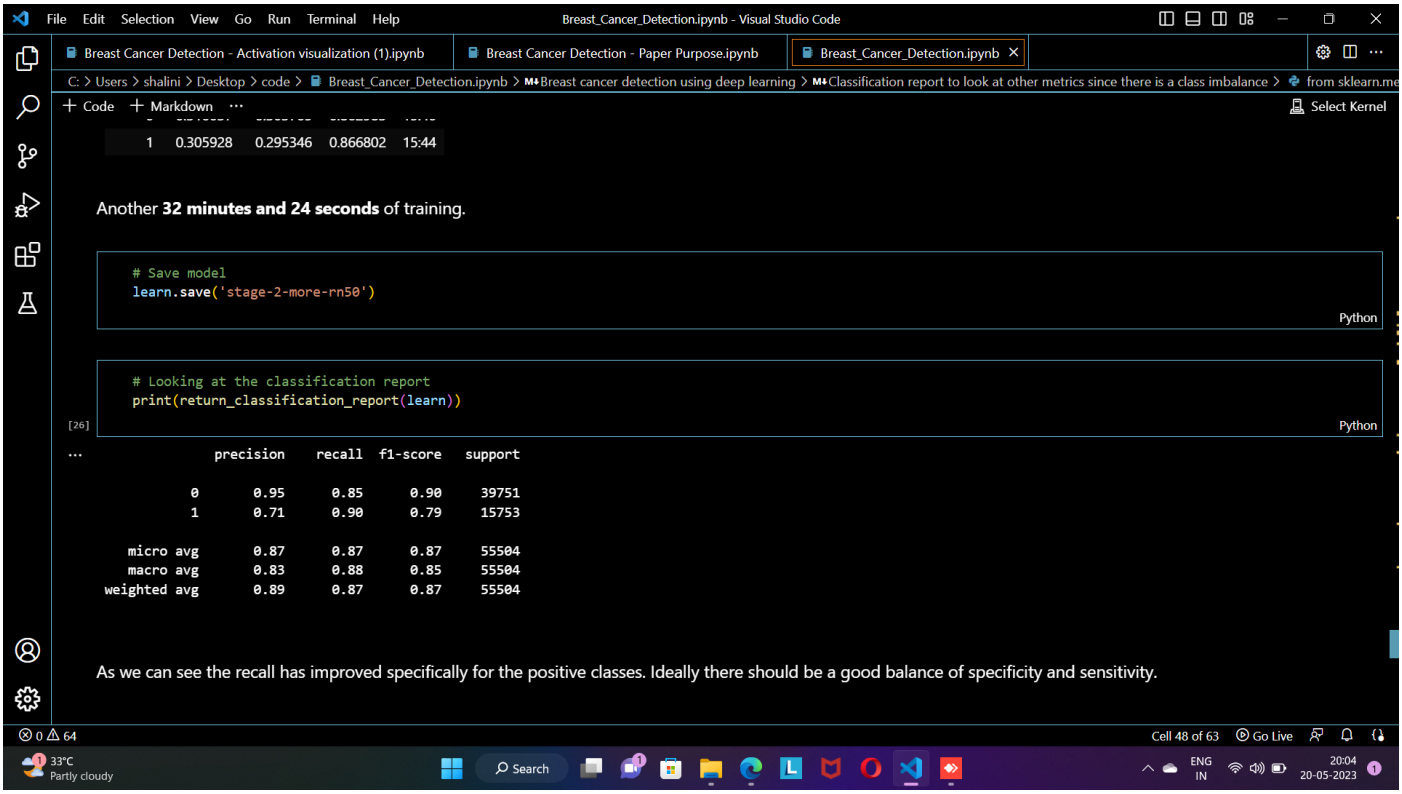
We now have a model which is **86.68%** accurate and has got a pretty **improved recall** for both in case of the +ve and the -ve classes.

We still could have trained the network for more. We have trained it for **7 epochs** and it took *approximately* **two hours.** More fine-tuning could have been done. Sophisticated data augmentation and resolution techniques could have been applied. But let's keep them aside for further studies for now :)

Model's architectural summary

```python
learn.summary()
```

```
==========================================================================
Layer (type)          Output Shape         Param #     Trainable
==========================================================================
Conv2d                [64, 24, 24]         9,408       True
BatchNorm2d           [64, 24, 24]         128         True
ReLU                  [64, 24, 24]         0           False
MaxPool2d             [64, 12, 12]         0           False
Conv2d                [64, 12, 12]         4,096       True
BatchNorm2d           [64, 12, 12]         128         True
Conv2d                [64, 12, 12]         36,864      True
BatchNorm2d           [64, 12, 12]         128         True
Conv2d                [256, 12, 12]        16,384      True
BatchNorm2d           [256, 12, 12]        512         True
```



DEMO:

https://youtu.be/UxnRPrdFwXg