

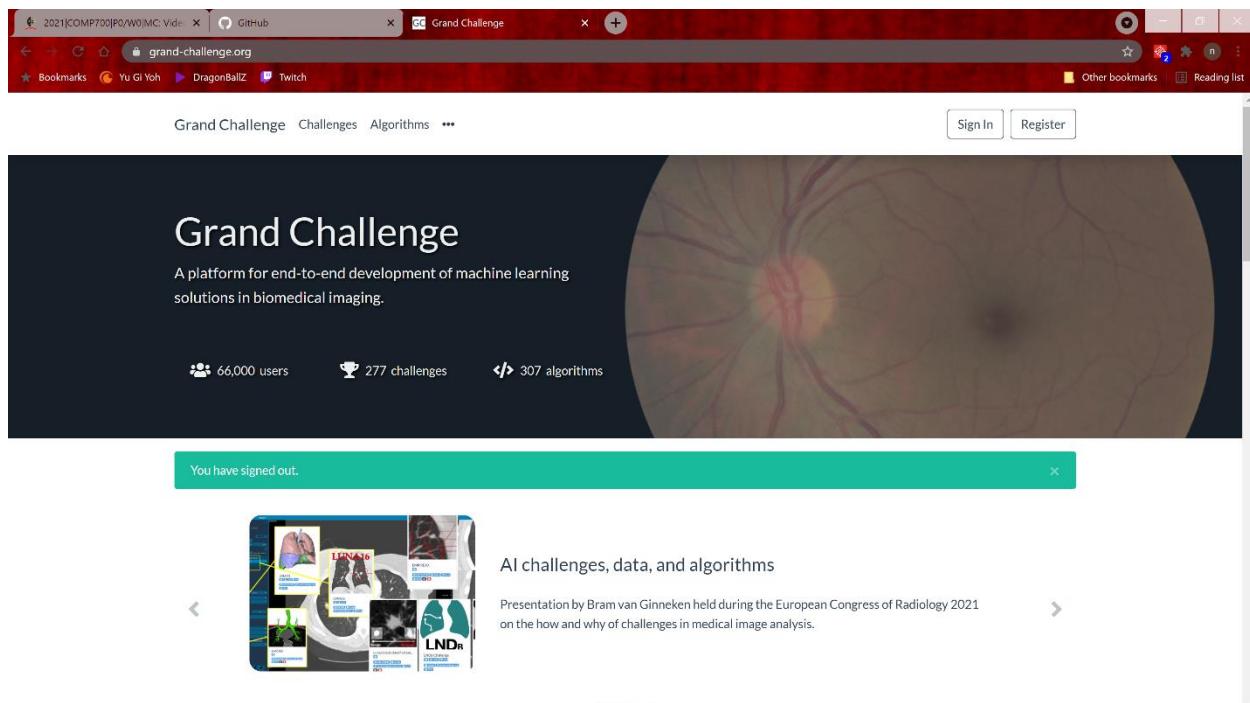
# How To Run The programs

## Clarifications

- The project assumes you will be using Pycharm and Google colab
- The project assumes you will upload relevant data to your google drive. Make sure to mount the google drive in runtimes for each notebook.
- The program assumes you will run 1 notebook at a time and disconnect the other when not in use.
- Pycharm code must be run before proceeding to the notebooks.
- Warning – google colab will terminate the session if it is left to idle for too long. Make sure to frequently check and move mouse at 10 minute intervals. Alternatively visit the site <https://stackoverflow.com/questions/57113226/how-to-prevent-google-colab-from-disconnecting> for information on how to stop colab from disconnecting.

## OBTAINING THE DATASET

The ICIAR 2018 dataset can be requested from the following site - <https://iciar2018-challenge.grand-challenge.org/Dataset/>



You will need to sign up before requesting the dataset

2021|COMP700|P0/W0/MC: Video GitHub Dataset - Grand Challenge

ic iar2018-challenge.grand-challenge.org/Dataset/ Bookmarks Yu Gi Yoh DragonBallZ Twitch Other bookmarks Reading list

Grand Challenge Challenges Algorithms ...

**Info** Submit Leaderboard Join

Home Rules Dataset Download Legacy-Submission Legacy-Results

BACH's dataset is publicly available under the CC BY-NC-ND license. The citation should refer to 10.1016/j.media.2019.05.010. Teams must notify the organisers of the challenge about any publication that is (partly) based on the results or data published on this site, in order to the challenge organization being able to maintain a list of publications associated with the challenge.

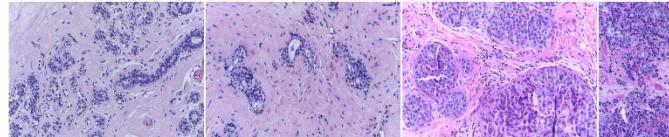
## Dataset

The dataset is composed of Hematoxylin and eosin (H&E) stained breast histology microscopy and whole-slide images. Challenge participants should evaluate the performance of their method on either/both sets of images.

[CLICK HERE TO REQUEST THE DATASET](#)

### 1. Microscopy images

Microscopy images are labelled as normal, benign, *in situ* carcinoma or invasive carcinoma according to the predominant cancer type in each image. The annotation was performed by two medical experts and images where there was disagreement were discarded.



Request the dataset and follow the prompts until you are in possession of the dataset. Once you have

2021|COMP700|P0/W0/MC: Video GitHub Dataset - Grand Challenge

ic iar2018-challenge.grand-challenge.org/download/ Bookmarks Yu Gi Yoh DragonBallZ Twitch Other bookmarks Reading list

Grand Challenge Challenges Algorithms ...

**Info** Submit Leaderboard Join

Home Rules Dataset Download Legacy-Submission Legacy-Results

BACH's dataset is publicly available under the CC BY-NC-ND license. The citation should refer to 10.1016/j.media.2019.05.010. Teams must notify the organisers of the challenge about any publication that is (partly) based on the results or data published on this site, in order to the challenge organization being able to maintain a list of publications associated with the challenge.



**ICIAR 2018 Grand Challenge on BreAst Cancer Histology images**

To download the dataset, please fill in [this form](#).

An e-mail with the link and access credentials to the dataset will be sent to your contact information.

Grand Challenge Policies Developers

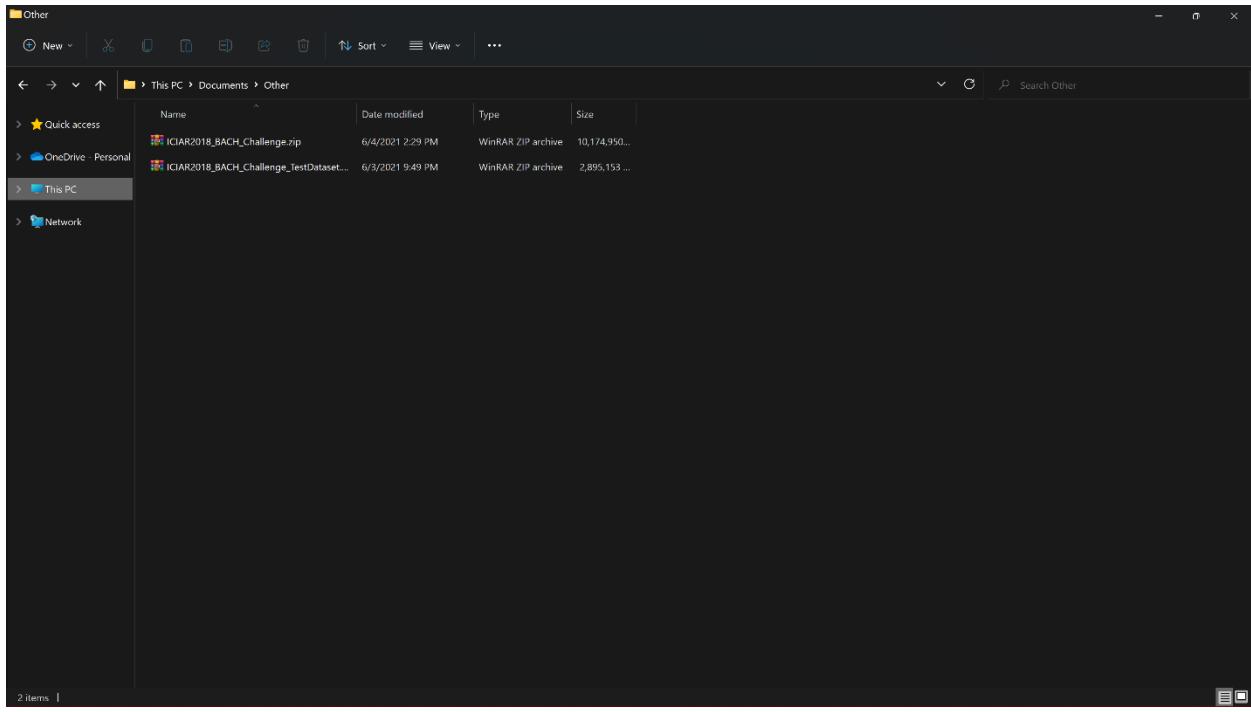
About Terms of Service API Documentation

Partners API Schema

Roadmap Developer Documentation

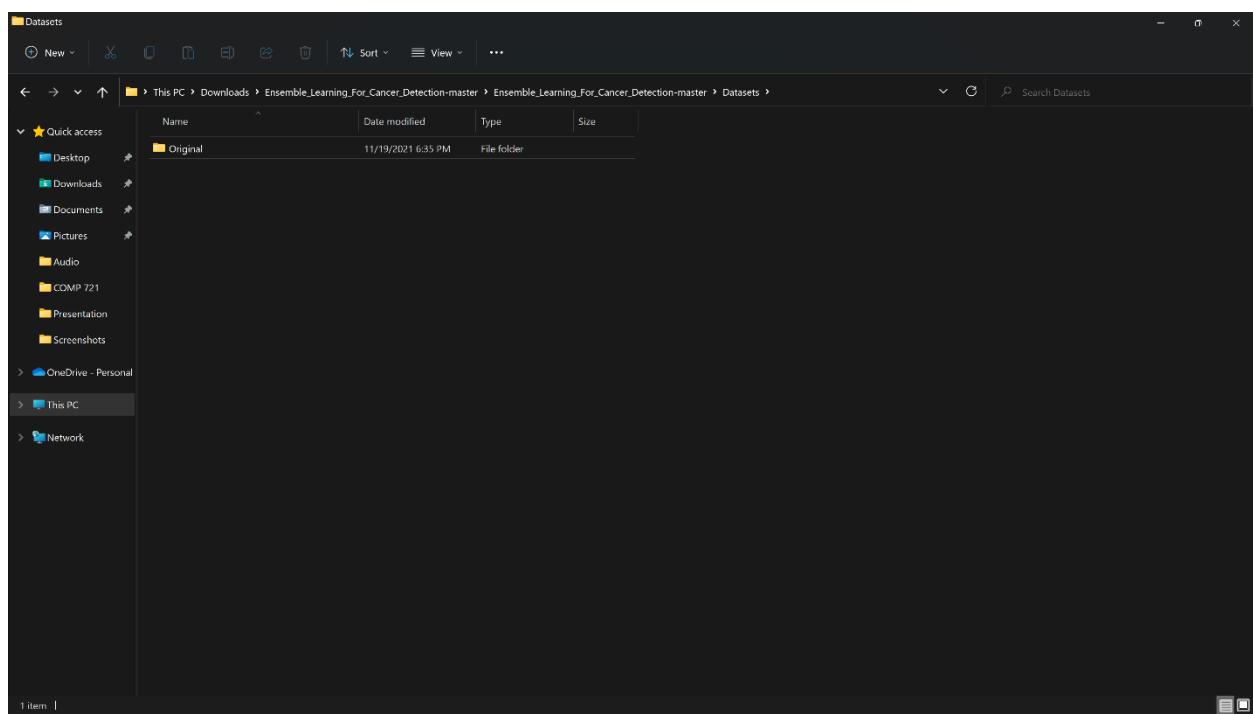
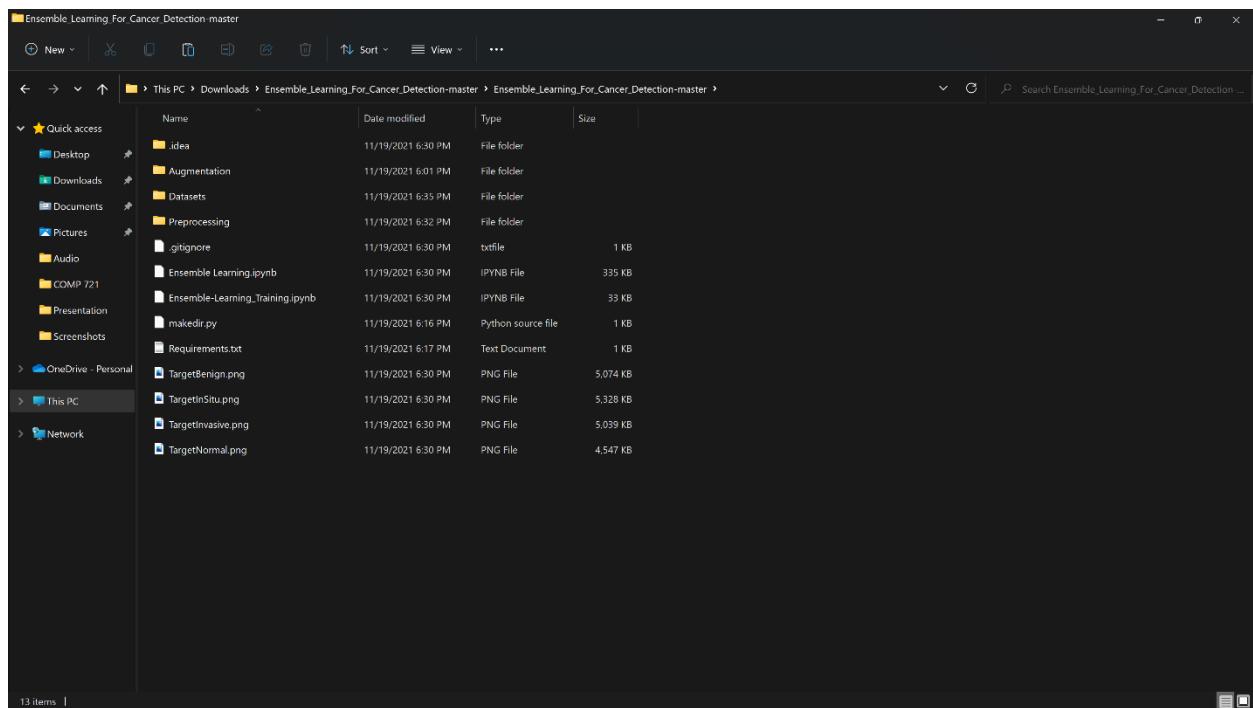
Support Statistics

© 2012-2021

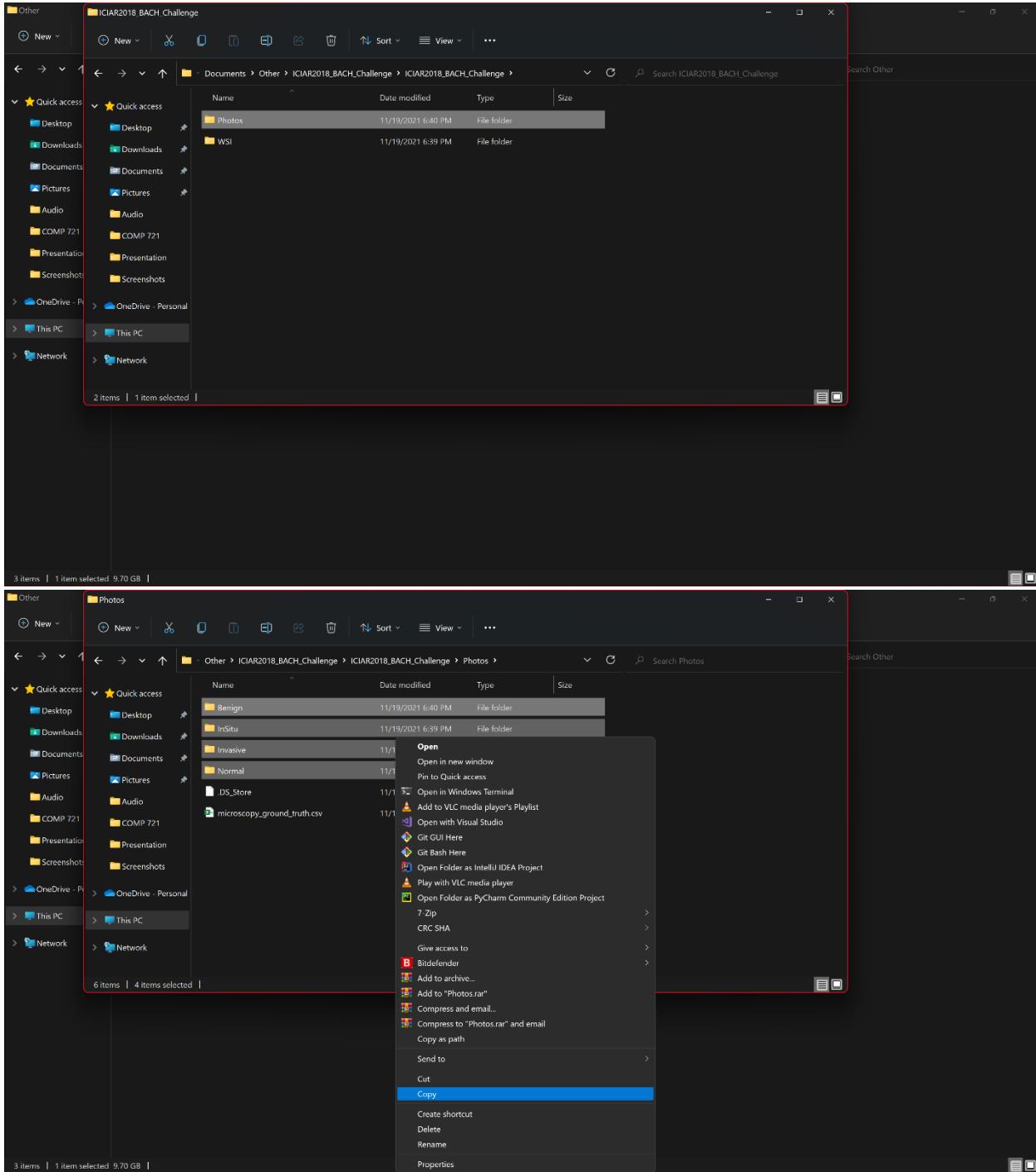


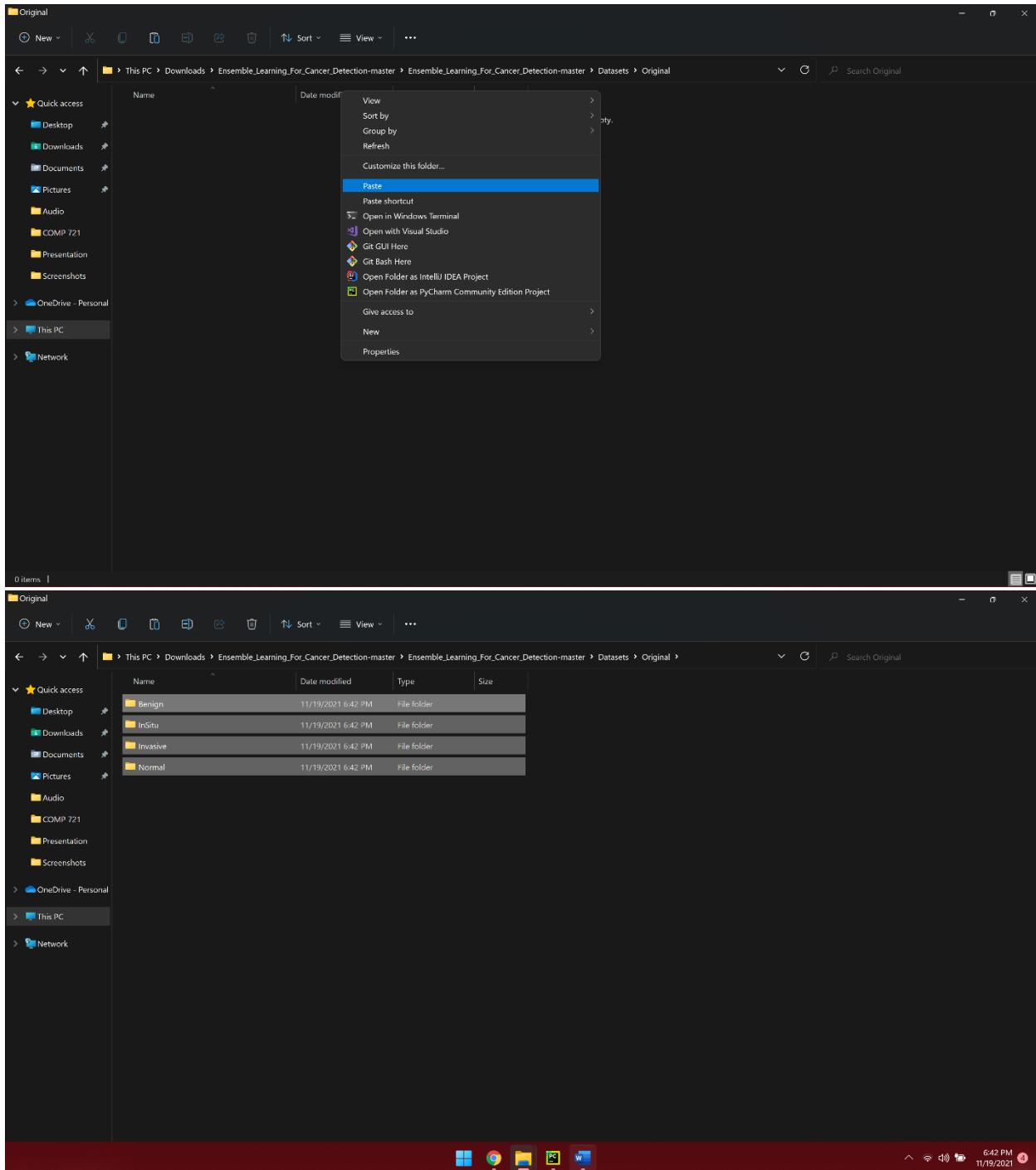
Once you are in possession of the dataset unzip the file names ICIAR2018\_BACH\_Challenge.zip . the other file may be disregarded.

Once you have obtained the code from github unzip it and create a folder called Datasets. Create another folder called Original inside of it



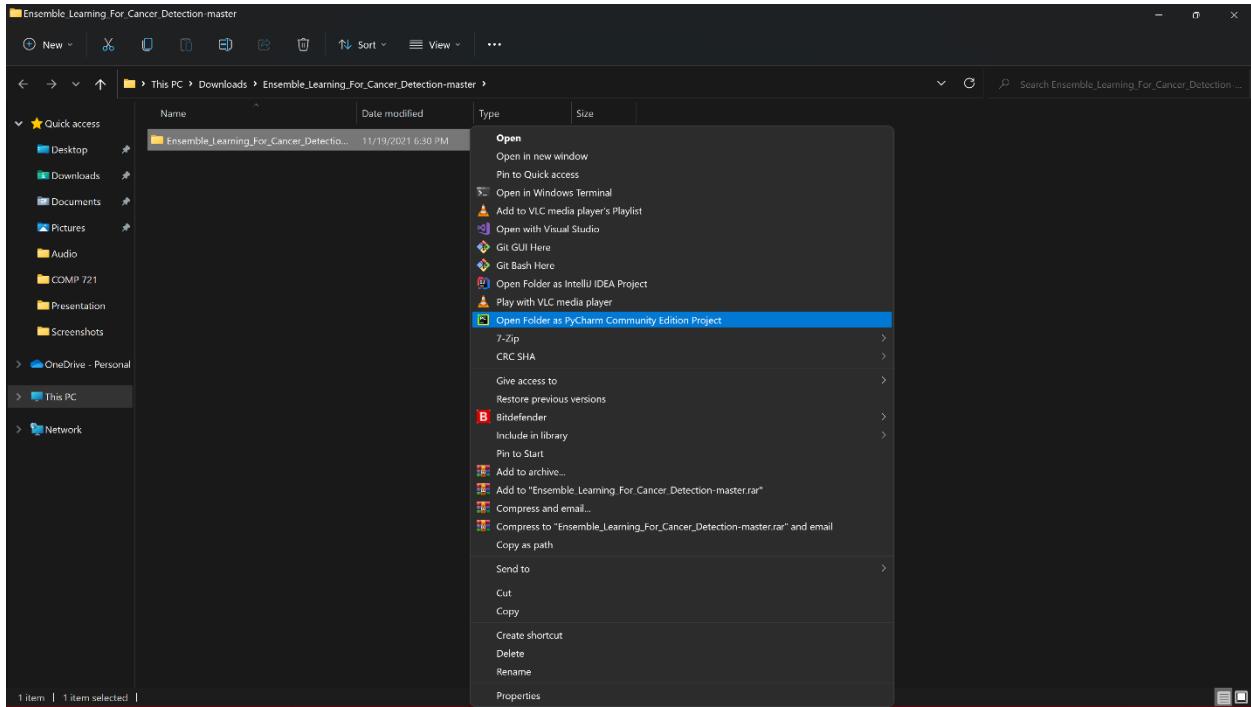
Open the unzipped ICIAR2018\_BACH\_Challenge and navigate until you see the following folders. Copy the folders named Benign, InSitu, Invasive and Normal and paste them in the folder Original that you created.



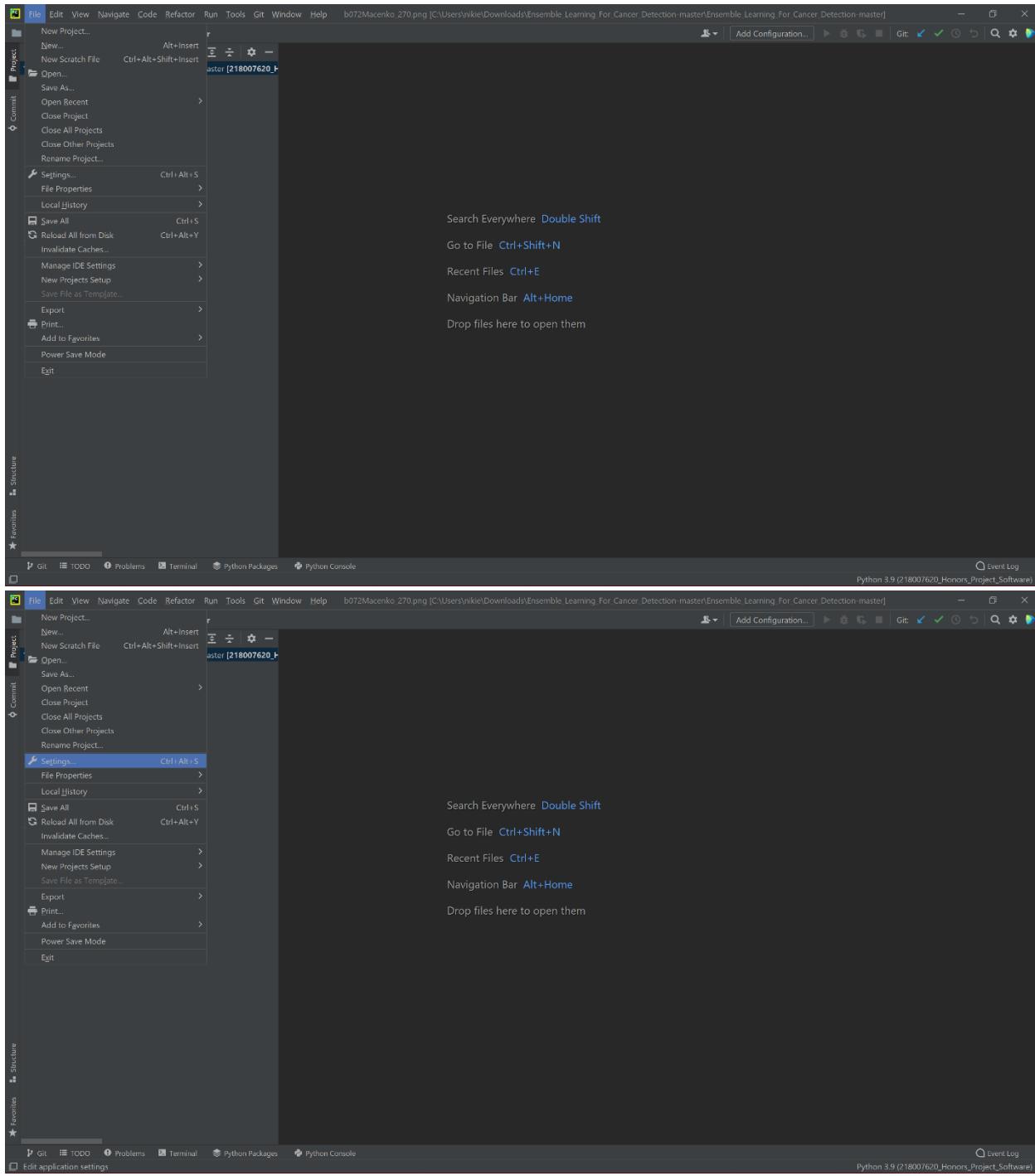


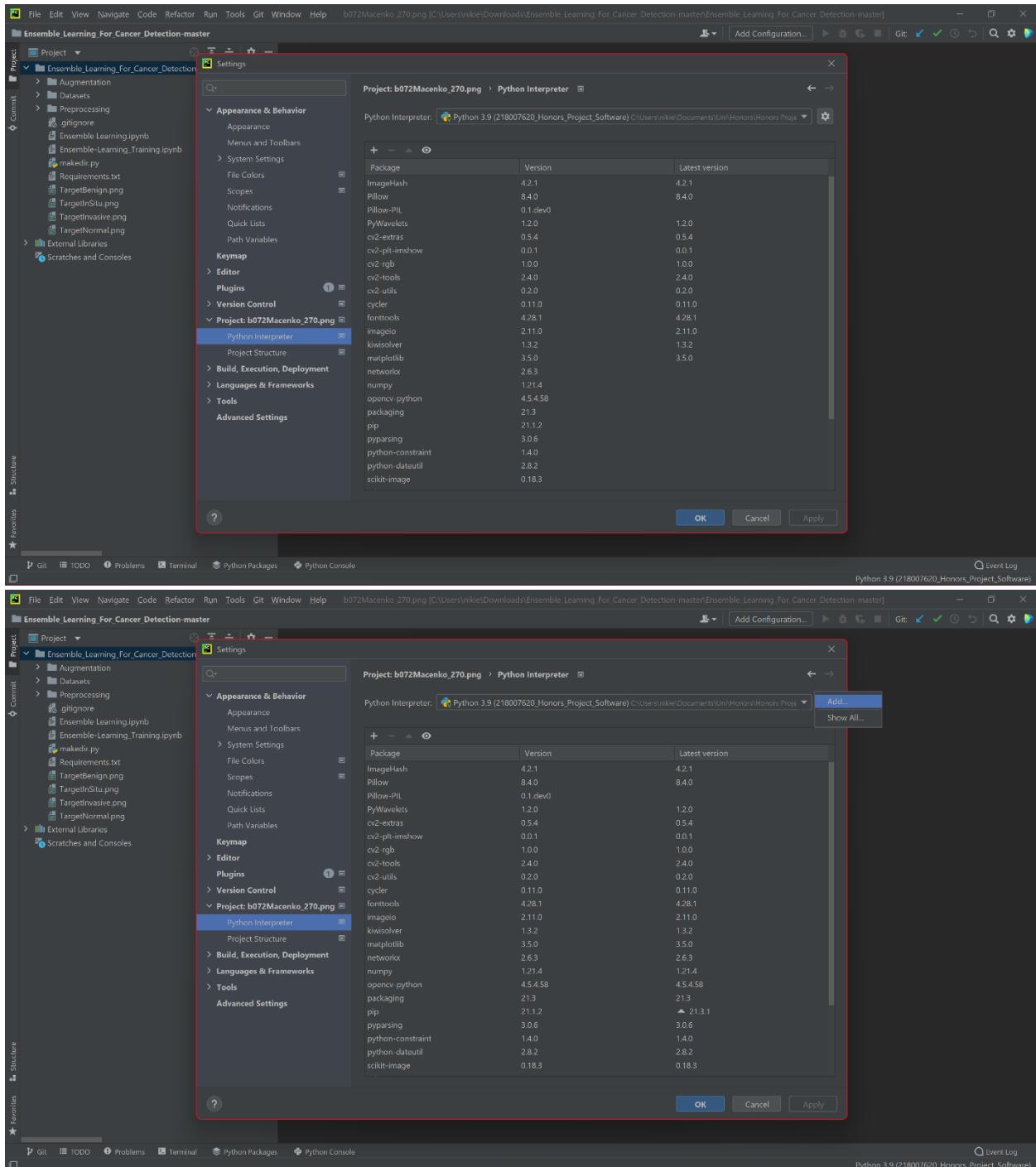
## SETTING UP VRITUAL ENVIROMENT ON PYCHARM

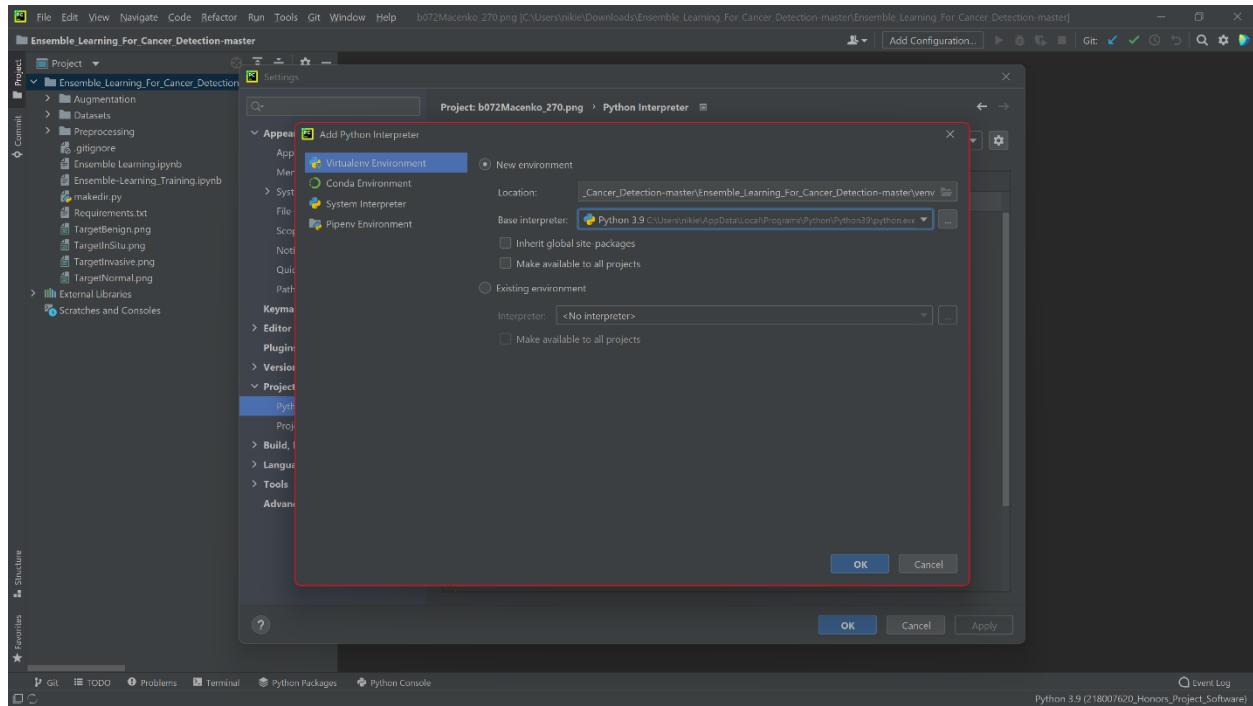
Once you have obtained the code from Github unzip the file and open up the folder in pycharm



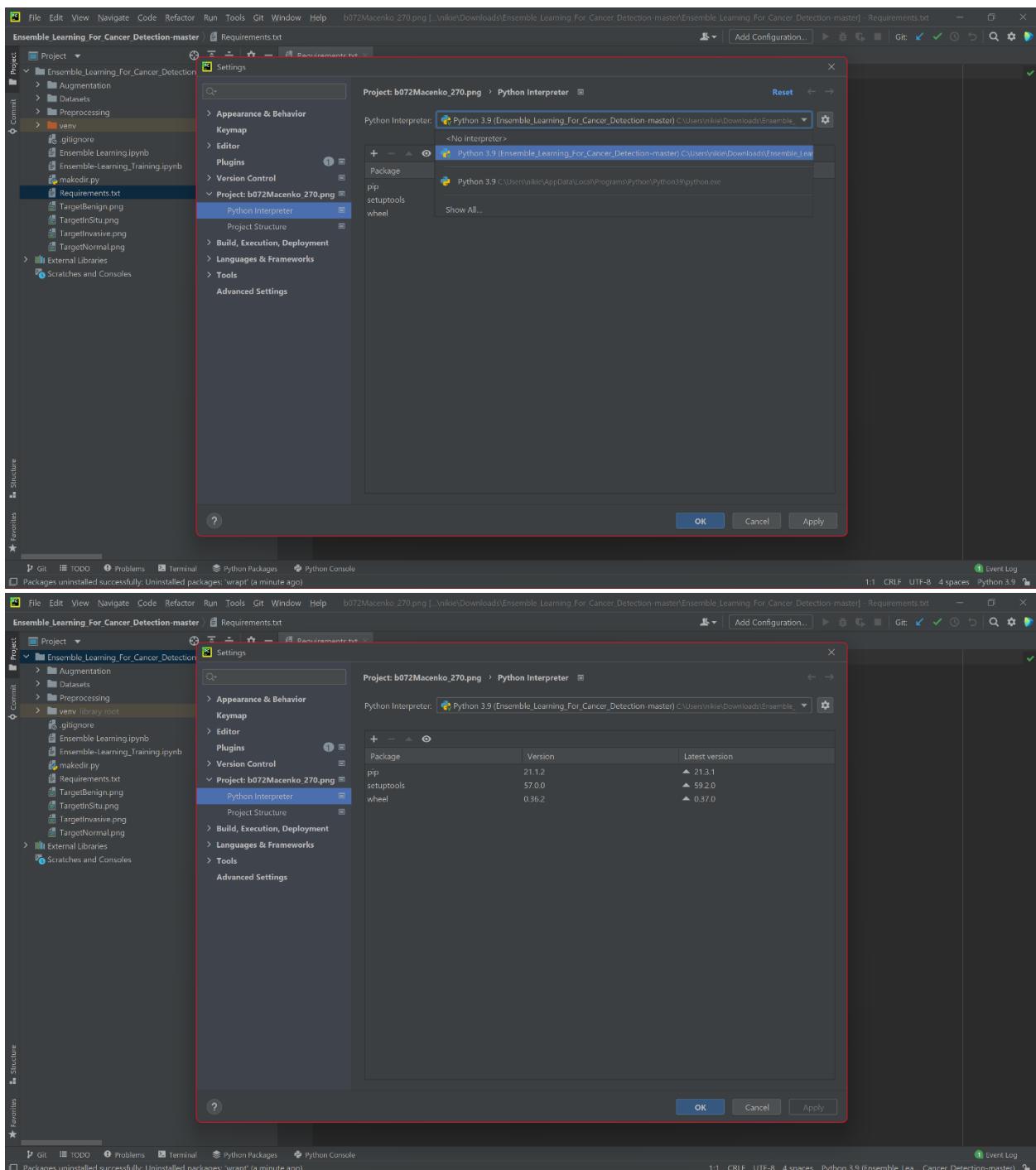
Once opened click File -> Settings -> Python Interpreter -> Add to add a new virtual environment







Choose your new python environment and click ok

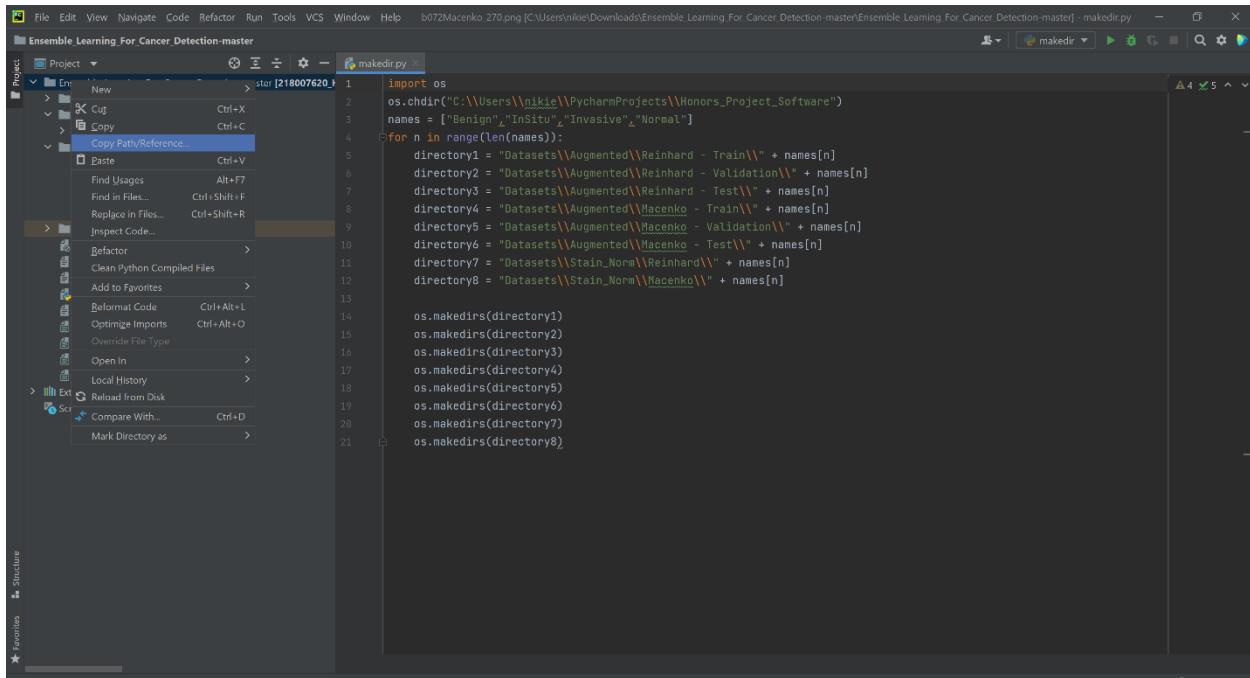


In the terminal run the following command -> pip install -r requirements.txt

Your Environment is now configured to run the programs.

## SETP 1 PREPROCESSING – STAIN NORMALIZATION AND AUGMENTATION.

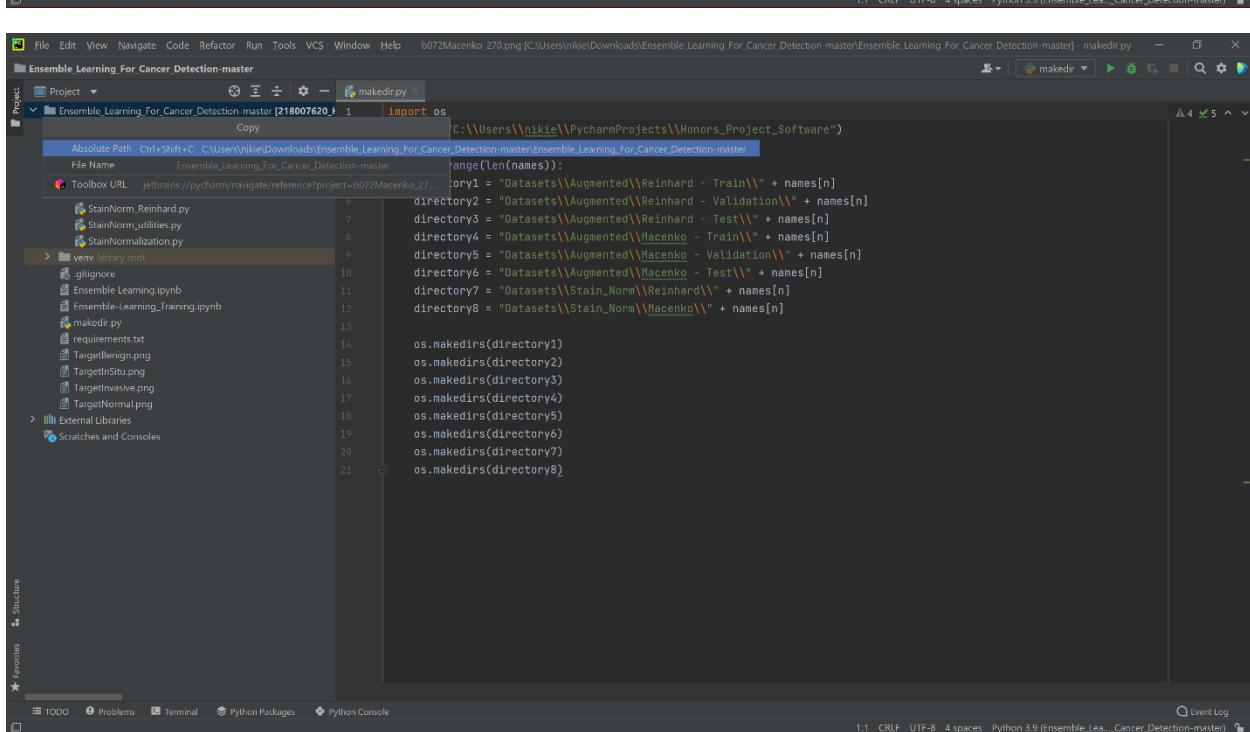
Change the line os.chdir to the current directory of your program. Following screenshot is a shortcut



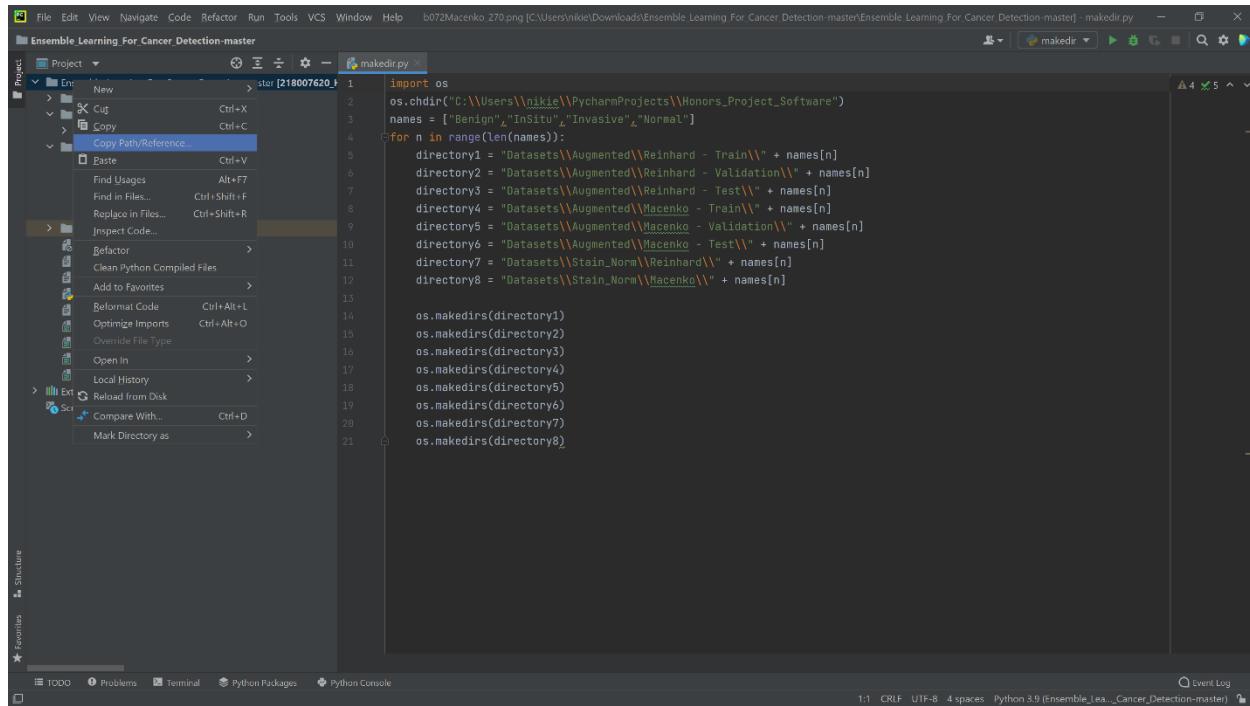
The screenshot shows the PyCharm IDE interface with the file 'makedir.py' open. A context menu is displayed over the line of code 'os.chdir("C:\\Users\\nikie\\PycharmProjects\\Honors\_Project\_Software")'. The menu includes options like 'Copy Path/Reference...', 'Paste', 'Find Usages', 'Find in Files...', 'Replace in Files...', 'Inspect Code...', 'Refactor', 'Clean Python Compiled Files', 'Add to Favorites', 'Reformat Code', 'Optimize Imports', 'Override File Type', 'Open In', 'Local History', 'Reload from Disk', 'Compare With...', and 'Mark Directory as'. The code itself is a script for creating directories for stain normalization and augmentation.

```
import os
os.chdir("C:\\Users\\nikie\\PycharmProjects\\Honors_Project_Software")
names = ["Benign", "InSitu", "Invasive", "Normal"]
for n in range(len(names)):
    directory1 = "Datasets\\Augmented\\Reinhard - Train\\" + names[n]
    directory2 = "Datasets\\Augmented\\Reinhard - Validation\\" + names[n]
    directory3 = "Datasets\\Augmented\\Reinhard - Test\\" + names[n]
    directory4 = "Datasets\\Augmented\\Macenko - Train\\" + names[n]
    directory5 = "Datasets\\Augmented\\Macenko - Validation\\" + names[n]
    directory6 = "Datasets\\Augmented\\Macenko - Test\\" + names[n]
    directory7 = "Datasets\\Stain_Norm\\Reinhard\\" + names[n]
    directory8 = "Datasets\\Stain_Norm\\Macenko\\" + names[n]

    os.makedirs(directory1)
    os.makedirs(directory2)
    os.makedirs(directory3)
    os.makedirs(directory4)
    os.makedirs(directory5)
    os.makedirs(directory6)
    os.makedirs(directory7)
    os.makedirs(directory8)
```

The screenshot shows the same PyCharm interface with the context menu still open over the 'os.chdir' line. The 'Copy' option under the 'Edit' menu has been selected, indicated by a blue highlight. The rest of the menu and the code editor are visible below.



File Edit View Navigate Code Refactor Run Tools VCS Window Help b072Macenko\_270.png [C:\Users\nikie\Downloads\Ensemble Learning For Cancer Detection-master\Ensemble Learning For Cancer Detection-master] - makedir.py

Project Ensemble Learning For Cancer Detection-master

makedir.py

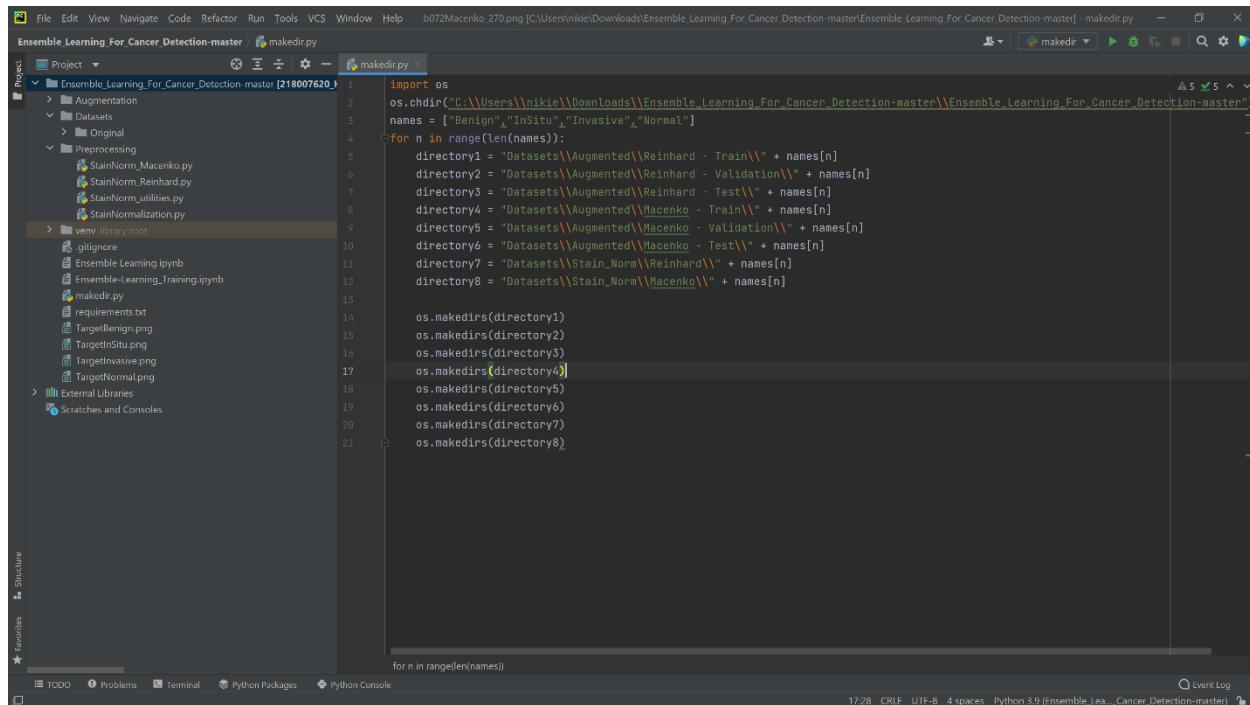
```
1 import os
2 os.chdir("C:\\\\Users\\\\nikie\\\\PycharmProjects\\\\Honors_Project_Software")
3 names = ["Benign", "InSitu", "Invasive", "Normal"]
4 for n in range(len(names)):
5     directory1 = "Datasets\\\\Augmented\\\\Reinhard - Train\\\\" + names[n]
6     directory2 = "Datasets\\\\Augmented\\\\Reinhard - Validation\\\\" + names[n]
7     directory3 = "Datasets\\\\Augmented\\\\Reinhard - Test\\\\" + names[n]
8     directory4 = "Datasets\\\\Augmented\\\\Macenko - Train\\\\" + names[n]
9     directory5 = "Datasets\\\\Augmented\\\\Macenko - Validation\\\\" + names[n]
10    directory6 = "Datasets\\\\Augmented\\\\Macenko - Test\\\\" + names[n]
11    directory7 = "Datasets\\\\Stain_Norm\\\\Reinhard\\\\" + names[n]
12    directory8 = "Datasets\\\\Stain_Norm\\\\Macenko\\\\" + names[n]
13
14    os.makedirs(directory1)
15    os.makedirs(directory2)
16    os.makedirs(directory3)
17    os.makedirs(directory4)
18    os.makedirs(directory5)
19    os.makedirs(directory6)
20    os.makedirs(directory7)
21    os.makedirs(directory8)
```

File Edit View Navigate Code Refactor Run Tools VCS Window Help b072Macenko\_270.png [C:\Users\nikie\Downloads\Ensemble Learning For Cancer Detection-master\Ensemble Learning For Cancer Detection-master] - makedir.py

Project Ensemble Learning For Cancer Detection-master

makedir.py

```
1 import os
2 os.chdir("C:\\\\Users\\\\nikie\\\\Downloads\\\\Ensemble_Learning_For_Cancer_Detection-master\\\\Ensemble_Learning_For_Cancer_Detection-master")
3 names = ["Benign", "InSitu", "Invasive", "Normal"]
4 for n in range(len(names)):
5     directory1 = "Datasets\\\\Augmented\\\\Reinhard - Train\\\\" + names[n]
6     directory2 = "Datasets\\\\Augmented\\\\Reinhard - Validation\\\\" + names[n]
7     directory3 = "Datasets\\\\Augmented\\\\Reinhard - Test\\\\" + names[n]
8     directory4 = "Datasets\\\\Augmented\\\\Macenko - Train\\\\" + names[n]
9     directory5 = "Datasets\\\\Augmented\\\\Macenko - Validation\\\\" + names[n]
10    directory6 = "Datasets\\\\Augmented\\\\Macenko - Test\\\\" + names[n]
11    directory7 = "Datasets\\\\Stain_Norm\\\\Reinhard\\\\" + names[n]
12    directory8 = "Datasets\\\\Stain_Norm\\\\Macenko\\\\" + names[n]
13
14    os.makedirs(directory1)
15    os.makedirs(directory2)
16    os.makedirs(directory3)
17    os.makedirs(directory4)
18    os.makedirs(directory5)
19    os.makedirs(directory6)
20    os.makedirs(directory7)
21    os.makedirs(directory8)
```



File Edit View Navigate Code Refactor Run Tools VCS Window Help b072Macenko\_270.png [C:\Users\nikie\Downloads\Ensemble Learning For Cancer Detection-master\Ensemble Learning For Cancer Detection-master] - makedir.py

Project Ensemble Learning For Cancer Detection-master

makedir.py

```
1 import os
2 os.chdir("C:\\\\Users\\\\nikie\\\\Downloads\\\\Ensemble_Learning_For_Cancer_Detection-master\\\\Ensemble_Learning_For_Cancer_Detection-master")
3 names = ["Benign", "InSitu", "Invasive", "Normal"]
4 for n in range(len(names)):
5     directory1 = "Datasets\\\\Augmented\\\\Reinhard - Train\\\\" + names[n]
6     directory2 = "Datasets\\\\Augmented\\\\Reinhard - Validation\\\\" + names[n]
7     directory3 = "Datasets\\\\Augmented\\\\Reinhard - Test\\\\" + names[n]
8     directory4 = "Datasets\\\\Augmented\\\\Macenko - Train\\\\" + names[n]
9     directory5 = "Datasets\\\\Augmented\\\\Macenko - Validation\\\\" + names[n]
10    directory6 = "Datasets\\\\Augmented\\\\Macenko - Test\\\\" + names[n]
11    directory7 = "Datasets\\\\Stain_Norm\\\\Reinhard\\\\" + names[n]
12    directory8 = "Datasets\\\\Stain_Norm\\\\Macenko\\\\" + names[n]
13
14    os.makedirs(directory1)
15    os.makedirs(directory2)
16    os.makedirs(directory3)
17    os.makedirs(directory4)
18    os.makedirs(directory5)
19    os.makedirs(directory6)
20    os.makedirs(directory7)
21    os.makedirs(directory8)
```

File Edit View Navigate Code Refactor Run Tools VCS Window Help b072Macenko\_270.png [C:\Users\nikie\Downloads\Ensemble Learning For Cancer Detection-master\Ensemble Learning For Cancer Detection-master] - makedir.py

Project Ensemble Learning For Cancer Detection-master

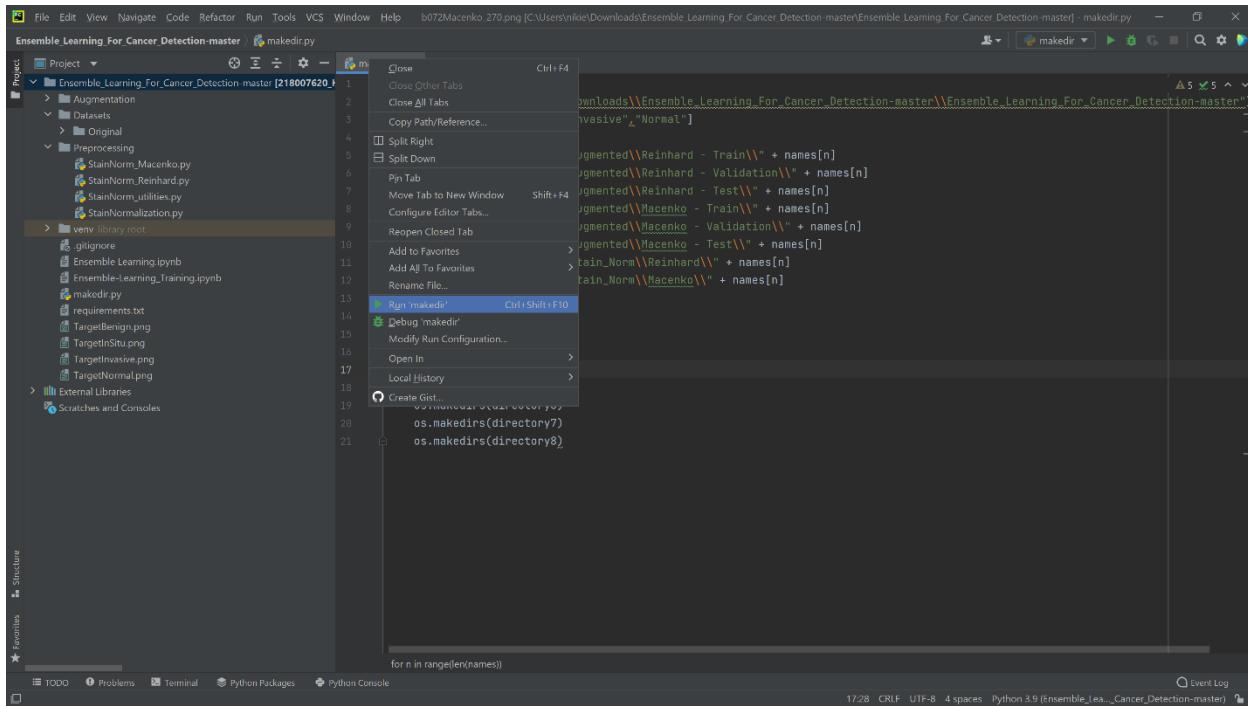
makedir.py

```
1 import os
2 os.chdir("C:\\\\Users\\\\nikie\\\\Downloads\\\\Ensemble_Learning_For_Cancer_Detection-master\\\\Ensemble_Learning_For_Cancer_Detection-master")
3 names = ["Benign", "InSitu", "Invasive", "Normal"]
4 for n in range(len(names)):
5     directory1 = "Datasets\\\\Augmented\\\\Reinhard - Train\\\\" + names[n]
6     directory2 = "Datasets\\\\Augmented\\\\Reinhard - Validation\\\\" + names[n]
7     directory3 = "Datasets\\\\Augmented\\\\Reinhard - Test\\\\" + names[n]
8     directory4 = "Datasets\\\\Augmented\\\\Macenko - Train\\\\" + names[n]
9     directory5 = "Datasets\\\\Augmented\\\\Macenko - Validation\\\\" + names[n]
10    directory6 = "Datasets\\\\Augmented\\\\Macenko - Test\\\\" + names[n]
11    directory7 = "Datasets\\\\Stain_Norm\\\\Reinhard\\\\" + names[n]
12    directory8 = "Datasets\\\\Stain_Norm\\\\Macenko\\\\" + names[n]
13
14    os.makedirs(directory1)
15    os.makedirs(directory2)
16    os.makedirs(directory3)
17    os.makedirs(directory4)
18    os.makedirs(directory5)
19    os.makedirs(directory6)
20    os.makedirs(directory7)
21    os.makedirs(directory8)
```

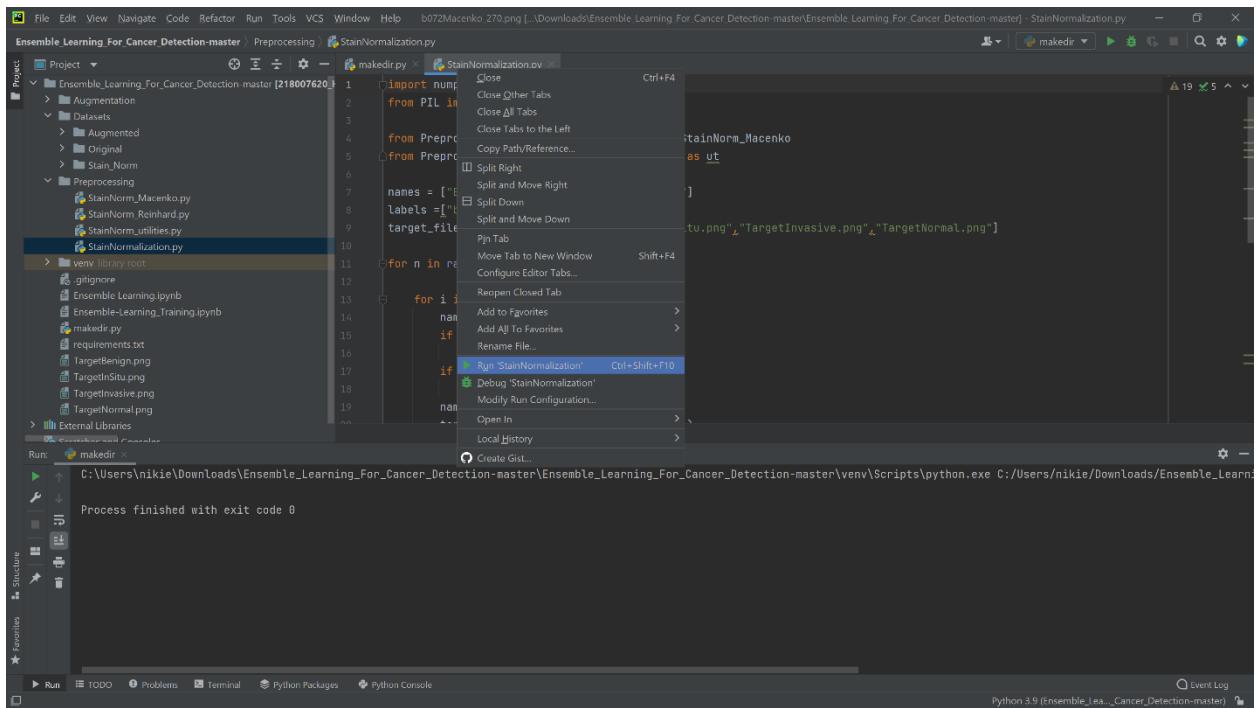
For safety when using directories always use "\\\"

Once you've made these changes

Run the file makedir.py first to create all necessary directories for reading and writing



Once all necessary directories are created you can process to run the StainNormalization.py file



Once The StainNormalization.py file has finished running. You can proceed to run the Augment\_images.py file

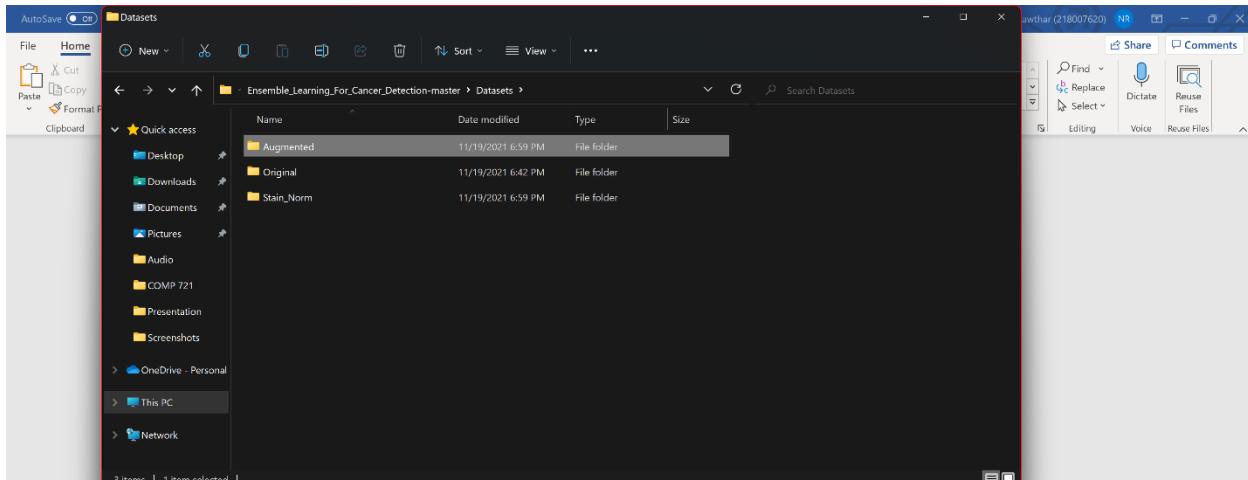
The screenshot shows the PyCharm IDE interface. The left sidebar displays a project structure for 'Ensemble Learning For Cancer Detection-master'. The main editor window contains the 'Augment\_images.py' file. A context menu is open over the code at line 12, with the 'Run' option highlighted. The code itself is a script for image augmentation, including imports for random and PIL, a rotation function, and a main loop for generating rotated images.

```
1 import random
2
3 from PIL import Image
4
5 def Rotation(image, dirname, isTrue, Stain):
6     num = random.randint(0, 3)
7     if isTrue:
8         for i in range(0, 360, 90):
9             if num * 90 == i:
10                 im = image.rotate(i).save("Test\\\\" + dirname + "\\\" + name + "_" + str(i) + ".png")
11             else:
12                 im = image.rotate(i).save("Validation\\\\" + dirname + "\\\" + name + "_" + str(i) + ".png")
13             print(name, Stain, isTrue, "Done")
14     else:
15         for i in range(0, 360, 90):
16             im = image.rotate(i).save("Train\\\\" + dirname + "\\\" + name + "_" + str(i) + ".png")
17     names = ["Benign", "InSitu", "Invasive"]
18     labels = ["b", "i", "iv"]
19
20     for n in range(len(names)):
21         setOfNumbers = set()
22         while len(setOfNumbers) < 15:
23             setOfNumbers.add(random.randint(1, 100))
24
25         for i in range(1, 101):
26             name = labels[n]
27             if i < 10:
28                 name += "00"
29             if i > 9 and i < 100:
30                 name += "0"
31             name += str(i)
32             Reinhard = name + "Reinhard"
33             Macenko = name + "Macenko"
```

You are now all set to begin training on Google Colab the next few steps will show you how to train a model and run the ensemble learning model.

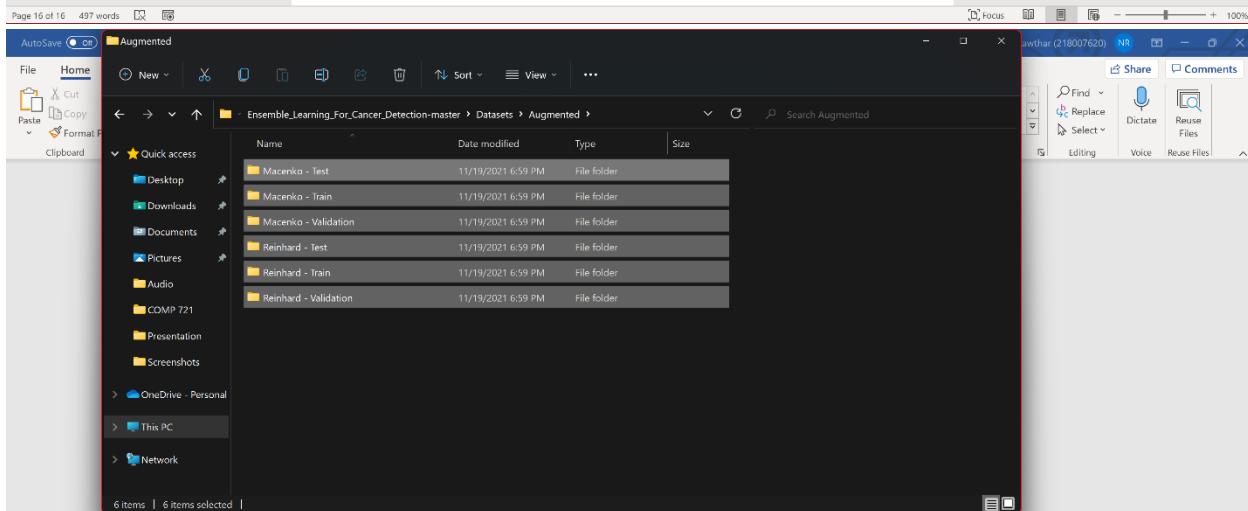
## STEP 2- TRAINING AND SAVING MODELS

Once you are done with step 1 upload your split dataset which can be found if you navigate from Datasets->Augmented-> here you will find 6 folders. If you pay for google drive and have more than 15Gb of space you can go ahead and upload all folders found in Dataset->Augment to your google drive. If you have the standard 15GB amount of space you can go ahead and upload either all Macenko folders or all Reinhard folders found in the Augmented folder.



#### STEP 2- TRAINING AND SAVING MODELS

Once you are done with step 1 upload your split dataset which can be found if you navigate from Datasets->Augmented-> here you will find 6 folders. If you pay for google drive and have more than 15Gb of space you can go ahead and upload all folders found in Dataset->Augment to your google drive. If you have the standard 15GB amount of space you can go ahead and upload either all Macenko folders or all Reinhard folders found in the Augmented folder.



#### STEP 2- TRAINING AND SAVING MODELS

Once you are done with step 1 upload your split dataset which can be found if you navigate from Datasets->Augmented-> here you will find 6 folders. If you pay for google drive and have more than 15Gb of space you can go ahead and upload all folders found in Dataset->Augment to your google drive. If you have the standard 15GB amount of space you can go ahead and upload either all Macenko folders or all Reinhard folders found in the Augmented folder.

Upload Ensemble-Learning\_Training.ipynb and Ensemble Learning.ipynb to your google drive and open them in colab

\*if you are using the Macenko dataset please update all names in the file i.e if something is named Reinhard change it to Macenko. This would've been automated but because you may change locations and naming structure this was the only possible way

Once you have opened Ensemble-Learning\_Training.ipynb and uploaded the datasets to Google Colab you can make a few changes before running. The following directories need to be changed to where you stored the dataset in your google drive.

Step A - The first highlighted text deals with where you would like to save your model and the name to associate it with. The default directory for google drive is /content/drive/MyDrive/. After /MyDrive/ you may either save the model as a .h5 file or enter the location along with a name for your saved model

\*Please save models using following guidelines best\_“Macenko/Reinhard”B”0-7 depending on the architecture you chose”.h5

Example if you're using the Macenko dataset and using efficientnet architecture B5

Best\_MacenckoB5.h5

The screenshot shows the Google Colab interface with the following details:

- Header:** Empower Students to Do Their Best, VAE.RECOMMENDER\_IMPLICIT, Colab Notebooks - Google Drive, Ensemble-Learning\_Training.ipynb
- Toolbar:** Bookmarks, Yu Gi Yoh, DragonBallZ, Twitch, Other bookmarks, Reading list
- File Menu:** File, Edit, View, Insert, Runtime, Tools, Help
- Code Cell 16-18:** Contains Python code for building a neural network model. It includes layers Dense(256, activation='ReLU'), Dropout(0.4), Flatten(), and Dense(4, activation='softmax').

```
16 model.add(layers.Dense(256, activation='ReLU'))
17 model.add(layers.Dropout(0.4))
18 # Set NUMBER_OF_CLASSES to the number of your final predictions.
19 model.add(layers.Flatten())
20 model.add(layers.Dense(4, activation="softmax"))
```
- Terminal Output:** Shows the download of weights from GitHub.

```
Downloading data from https://github.com/Callidior/keras-applications/releases/download/efficientnet/efficientnet-b0_weights_tf_dim_ordering_tf_kernels_autoaugment_notop.h5
16809984/16804768 [=====] - 1s 0us/step
16818176/16804768 [=====] - 1s 0us/step
```
- Code Cell 19-20:** Contains code for training the model using ReduceLROnPlateau and EarlyStopping callbacks, and saving checkpoints.

```
1 batch_size = 16
2
3 reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=2, min_lr=0.0001, mode='max')
4 earlyStop = EarlyStopping(monitor='/content/drive/MyDrive/218007620_HonorsProject/Model/best_MacenckoB0.h5')
5 modelCheckpoint = ModelCheckpoint('/content/drive/MyDrive/218007620_HonorsProject/Model/best_MacenckoB0.h5', monitor='val_acc',
6                                     mode='max', save_weights_only=False, save_best_only=True)
7
8
9 TrainGen = datagen.flow_from_directory( "/content/drive/MyDrive/218007620_HonorsProject/Macenko - Train",
10                                         target_size=(224, 224), color_mode="rgb", class_mode="categorical",
11                                         batch_size=batch_size, shuffle=True, seed=15)
12
13 ValGen = valgen.flow_from_directory( "/content/drive/MyDrive/218007620_HonorsProject/Macenko - Validation",
14                                         target_size=(224, 224), color_mode="rgb", class_mode="categorical",
15                                         batch_size=batch_size, shuffle=False, seed=15)
16
17 TestGen = testgen.flow_from_directory("/content/drive/MyDrive/218007620_HonorsProject/Macenko - Test",
18                                         target_size=(224, 224), color_mode = "rgb", batch_size = 1, seed=15, shuffle=False)
```
- Terminal Log:** Shows the count of images found in each class.

```
Found 1360 images belonging to 4 classes.
Found 180 images belonging to 4 classes.
Found 60 images belonging to 4 classes.
```

Step B - The second, third and fourth highlighted texts deal with the location of the training, validation and testing data you can go ahead and change these to the location you have them stored in google drive.

\*note /content/drive/MyDrive/ is the root and needs to be added when accessing files

The screenshot shows a Google Colab notebook titled "Ensemble-Learning\_Training.ipynb". The code in the cell is as follows:

```
16 model.add(layers.Dense(256, activation='ReLU'))
17 model.add(layers.Dropout(0.4))
18 # Set NUMBER_OF_CLASSES to the number of your final predictions.
19 model.add(layers.Flatten())
20 model.add(layers.Dense(4, activation="softmax"))

Downloading data from https://github.com/Callidior/keras-applications/releases/download/efficientnet/efficientnet-b0_weights_tf_dim_ordering_tf_kernels_autoaugment_notop.h5
16809984/16804768 [=====] - 1s 0us/step
16818176/16804768 [=====] - 1s 0us/step

1 batch_size = 16
2
3 reduceLr = ReduceLROnPlateau(monitor='val_loss',factor=0.2, patience=8, min_lr=0.00001, mode='max')
4 earlyStop = EarlyStopping(monitor='val_loss', mode = 'min', patience=10)
5 modelCheckpoint = ModelCheckpoint('/content/drive/MyDrive/218007620_HonorsProject/Model/best_MacenkoB0.h5',monitor='val_acc',
6                                     mode='max', save_weights_only=False, save_best_only=True)
7
8
9 TrainGen = datagen.flow_from_directory('/content/drive/MyDrive/218007620_HonorsProject/Macenko - Train',
10                                         target_size=(224, 224), color_mode='rgb',class_mode='categorical',
11                                         batch_size=batch_size, shuffle=True, seed=15)
12
13 ValGen = valgen.flow_from_directory('/content/drive/MyDrive/218007620_HonorsProject/Macenko - Validation',
14                                         target_size=(224, 224), color_mode='rgb',class_mode='categorical',
15                                         batch_size=batch_size, shuffle=False, seed=15)
16
17 TestGen = testgen.flow_from_directory('/content/drive/MyDrive/218007620_HonorsProject/Macenko - Test',
18                                         target_size=(224, 224),color_mode = 'rgb',batch_size = 1,seed=15, shuffle=False)

Found 1360 images belonging to 4 classes.
Found 180 images belonging to 4 classes.
Found 60 images belonging to 4 classes.
```

The screenshot shows the same Google Colab notebook "Ensemble-Learning\_Training.ipynb". The code is identical to the one in the previous screenshot. A cursor is hovering over the text "ctrl + click" in the code, which is part of a tooltip or a specific code comment. The rest of the notebook interface is visible, including the toolbar and other tabs.

```

16 model.add(layers.Dense(256, activation="ReLU"))
17 model.add(layers.Dropout(0.4))
18 # Set NUMBER_OF_CLASSES to the number of your final predictions.
19 model.add(layers.Flatten())
20 model.add(layers.Dense(4, activation="softmax"))

Downloading data from https://github.com/Callidior/keras_applications/releases/download/efficientnet/efficientnet-b0_weights_tf_dim_ordering_tf_kernels_autoaugment_notop.h5
16809984/16804768 [=====] - 1s 0us/step
16818176/16804768 [=====] - 1s 0us/step

1 batch_size = 16
2
3 reduceLr = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=8, min_lr=0.00001, mode='max')
4 earlyStop = EarlyStopping(monitor='val_loss', mode = 'min', patience=10)
5 modelCheckpoint = ModelCheckpoint("/content/drive/MyDrive/218007620 HonorsProject/Model/best_MacenkoB0.h5", monitor='val_acc',
6                                     mode='max', save_weights_only=False, save_best_only=True)
7
8
9 TrainGen = datagen.flow_from_directory( "/content/drive/MyDrive/218007620 HonorsProject/Macenko - Train",
10                                         target_size=(224, 224), color_mode='rgb', class_mode='categorical',
11                                         batch_size=batch_size, shuffle=True, seed=15)
12
13 ValGen = valgen.flow_from_directory( "/content/drive/MyDrive/218007620 HonorsProject/Macenko - Validation",
14                                         target_size=(224, 224), color_mode='rgb', class_mode='categorical',
15                                         batch_size=batch_size, shuffle=True, seed=15)
16
17 TestGen = testgen.flow_from_directory( "/content/drive/MyDrive/218007620 HonorsProject/Macenko - Test",
18                                         target_size=(224, 224), color_mode = 'rgb', batch_size = 1, seed=15, shuffle=False)

Found 1360 images belonging to 4 classes.
Found 188 images belonging to 4 classes.
Found 60 images belonging to 4 classes.

```

The last adjustment that needs to be made is again where you are saving the model in Step A- go ahead and copy that directory and past it here.

```

1 from sklearn.metrics import confusion_matrix
2
3 import numpy as np
4
5 y_pred = model.predict(TestGen, verbose=1)
6
7 y_pred = np.argmax(y_pred, axis = 1)
8
9
10 print(confusion_matrix(TestGen.classes, y_pred))
11
12 print(classification_report(TestGen.classes, y_pred))
13
14 model.save("/content/drive/MyDrive/Model/best_MacenkoB0.h5")

60/60 [=====] - 97s 2s/step
[[1  2  1  1]
 [ 0 14  1  0]
 [ 0  1 14  0]
 [ 0  0  0 15]]
 precision    recall   f1-score   support
      0       1.00      0.73      0.85      15
      1       0.82      0.93      0.87      15
      2       0.88      0.93      0.90      15
      3       0.94      1.00      0.97      15

   accuracy          0.90      0.90      0.90      60
  macro avg       0.91      0.90      0.90      60
 weighted avg     0.91      0.90      0.90      60

```

\*Note there are different EfficientNets architectures. Each with a different img\_size required. Depending on the model you choose change the img\_size variable in block 4 to the corresponding size

B0	224
B1	240
B2	260
B3	300
B4	380
B5	456
B6	528
B7	600

The final change is the selection of what model you want to train. On line 7 of code block 3 the number after B can be changed so you can choose different architectures to train. The range is 0-7 and remember to update the img\_size to the corresponding image size needed for each architecture.

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** Empower Students to Do Their Best! > VAE.RECOMMENDER.IMPPLICIT... > Colab Notebooks - Google Drive > Ensemble-Learning.Training.ipynb
- Toolbar:** Bookmarks, Yu Gi Yoh, DragonBallZ, Twitch, Other bookmarks, Reading list.
- Header:** Ensemble-Learning\_Training.ipynb, File Edit View Insert Runtime Tools Help All changes saved, Connect, Share, Editing.
- Code Cell 1:** Python code for data generators and model architecture using TensorFlow and EfficientNet.

```
[ ] 1 datagen = ImageDataGenerator(rescale=1.0/255, rotation_range=5, width_shift_range=0.1, height_shift_range = 0.1,
2                                     zoom_range= 0.3, horizontal_flip=True, vertical_flip=True, fill_mode='reflect')
3 valgen = ImageDataGenerator(rescale = 1.0/255)
4
5 testgen = ImageDataGenerator(rescale = 1.0/255)
6
7 conv_base = efn.EfficientNetB0(weights="imagenet", include_top=False )
8
9 model = models.Sequential()
10 model.add(conv_base)
11
12 model.add(layers.GlobalAveragePooling2D(name="gap"))
13 model.add(layers.Flatten())
14 #void overfitting
15 model.add(layers.Dropout(0.5))
16 model.add(layers.Dense(256, activation="ReLU"))
17 model.add(layers.Dropout(0.4))
18 # Set NUMBER_OF_CLASSES to the number of your final predictions.
19 model.add(layers.Flatten())
20 model.add(layers.Dense(4, activation="softmax"))
```
- Code Cell 2:** Python code defining batch size and image size.

```
[ ] 1 batch_size = 16
2 img_size = 224
3
```

Once all these changes are made connect to a run time and click run all

```
+ Code + Text
[ ] 6 from tensorflow.keras.callbacks import ModelCheckpoint
[ ] 7 import pickle
[ ] 8 from tensorflow.keras.optimizers import Adam
[ ] 9 from sklearn.metrics import accuracy_score, classification_report
[ ] 10
[ ] 11 !pip install -U efficientnet
[ ] 12 import efficientnet.tfkeras as efn

[ ] 1 datagen = ImageDataGenerator(rescale=1.0/255,rotation_range=5, width_shift_range=0.1, height_shift_range = 0.1,
[ ] 2           zoom_range= 0.3, horizontal_flip=True, vertical_flip=True, fill_mode='reflect')
[ ] 3 valgen = ImageDataGenerator(rescale = 1.0/255)
[ ] 4
[ ] 5 testgen = ImageDataGenerator(rescale = 1.0/255)
[ ] 6
[ ] 7 conv_base = efn.EfficientNetB0(weights="imagenet", include_top=False )
[ ] 8
[ ] 9 model = models.Sequential()
[ ] 10 model.add(conv_base)
[ ] 11
[ ] 12 model.add(layers.GlobalAveragePooling2D(name="gap"))
[ ] 13 model.add(layers.Flatten())
[ ] 14 #void overfitting
[ ] 15 model.add(layers.Dropout(0.5))
[ ] 16 model.add(layers.Dense(256, activation="ReLU"))
[ ] 17 model.add(layers.Dropout(0.4))
[ ] 18 # Set NUMBER_OF_CLASSES to the number of your final predictions.
[ ] 19 model.add(layers.Flatten())
[ ] 20 model.add(layers.Dense(4, activation="softmax"))

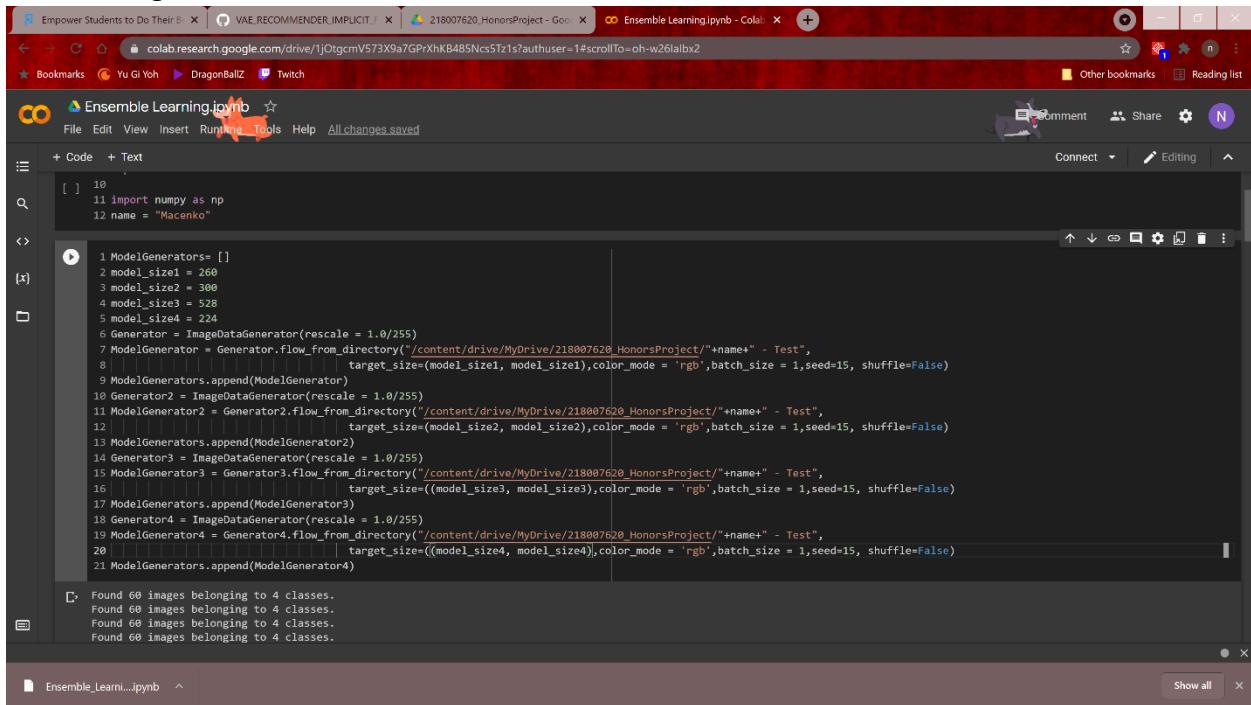
[ ] 1 batch_size = 16
[ ] 2 img_size = 224
[ ] 3
```

```
+ Code + Text Run all Ctrl+F9
[ ] 6 from tensorflow:
7 import pickle
8 from tensorflow:
9 import sklearn.
10
11 !pip install
12 import efficientnet
[ ] 1 datagen = ImageDataGenerator(
2
3 valgen = ImageDataGenerator(
4
5 testgen = ImageDataGenerator(
6
7 conv_base = conv_base
8
9 model = model
10 model.add(conv_base)
11
12 model.add(layers.GlobalAveragePooling2D(name="gap"))
13 model.add(layers.Flatten())
14 #void overfitting
15 model.add(layers.Dropout(0.5))
16 model.add(layers.Dense(256, activation="ReLU"))
17 model.add(layers.Dropout(0.4))
18 # Set NUMBER_OF_CLASSES to the number of your final predictions.
19 model.add(layers.Flatten())
20 model.add(layers.Dense(4, activation="softmax"))

[ ] 1 batch_size = 16
2 img_size = 224
3
```

Once you have done training a minimum of 4 different models you may proceed to use the Ensemble Learning.ipynb file

A few changes need to be made here as well



The screenshot shows a Google Colab interface with the title "Ensemble Learning.ipynb". The code cell contains Python code for generating four ModelGenerators from different directories in Google Drive. The code includes imports for numpy, defines model sizes (model\_size1=260, model\_size2=300, model\_size3=528, model\_size4=224), and creates generators for each. The generators flow from specific directories in "MyDrive/218007620\_HonorsProject" with target sizes matching the model sizes. The output pane shows a log message indicating 60 images found per class for all four models.

```
[ ] 10
11 import numpy as np
12 name = "Macenko"
13
14 ModelGenerators= []
15 model_size1 = 260
16 model_size2 = 300
17 model_size3 = 528
18 model_size4 = 224
19
20 Generator = ImageDataGenerator(rescale = 1.0/255)
21 ModelGenerator = Generator.flow_from_directory("/content/drive/MyDrive/218007620_HonorsProject/" + name + " - Test",
22                                         target_size=(model_size1, model_size1),color_mode = 'rgb',batch_size = 1,seed=15, shuffle=False)
23 ModelGenerators.append(ModelGenerator)
24 Generator2 = ImageDataGenerator(rescale = 1.0/255)
25 ModelGenerator2 = Generator2.flow_from_directory("/content/drive/MyDrive/218007620_HonorsProject/" + name + " - Test",
26                                         target_size=(model_size2, model_size2),color_mode = 'rgb',batch_size = 1,seed=15, shuffle=False)
27 ModelGenerators.append(Modelgenerator2)
28 Generator3 = ImageDataGenerator(rescale = 1.0/255)
29 ModelGenerator3 = Generator3.flow_from_directory("/content/drive/MyDrive/218007620_HonorsProject/" + name + " - Test",
30                                         target_size=((model_size3, model_size3)),color_mode = 'rgb',batch_size = 1,seed=15, shuffle=False)
31 ModelGenerators.append(ModelGenerator3)
32 Generator4 = ImageDataGenerator(rescale = 1.0/255)
33 ModelGenerator4 = Generator4.flow_from_directory("/content/drive/MyDrive/218007620_HonorsProject/" + name + " - Test",
34                                         target_size=((model_size4, model_size4)),color_mode = 'rgb',batch_size = 1,seed=15, shuffle=False)
35 ModelGenerators.append(Modelgenerator4)
36
37 ModelGenerators[0].next()
38 ModelGenerators[1].next()
39 ModelGenerators[2].next()
40 ModelGenerators[3].next()
41
42 Found 60 images belonging to 4 classes.
43 Found 60 images belonging to 4 classes.
44 Found 60 images belonging to 4 classes.
45 Found 60 images belonging to 4 classes.
```

Firstly for each ModelGenerator please pass through the location you have placed the test set in on google drive. Please try to remember the model you plan on using and update the model\_size variable to the corresponding efficientnets architecture used. If you plan on using the B5 model for ModelGenerator2 then change the size for model generator 2 to the appropriate size.

The final change you need to make is the location you stored your models. Once again this is dependent on the Model Generator sizes you updated. The program will perform poorly if it does not get the required target size variable. For each model in the below code update the location of each of your chosen 4 models. Once you have updated this you may click connect and run all just like you did in the previous notebook

The screenshot shows a Google Colab interface with the title "Ensemble Learning.ipynb". The code cell contains Python code for loading multiple pre-trained models and creating an ensemble estimator. The output cell shows the results of running the code, indicating that 60 images were found belonging to 4 classes.

```
17 ModelGenerators.append(ModelGenerator3)
18 Generator4 = ImageDataGenerator(rescale = 1.0/255)
19 ModelGenerator4 = Generator4.flow_from_directory('/content/drive/MyDrive/218007620_HonorsProject/' + name + ' - Test',
20                                                 target_size=(model_size[0], model_size[1]), color_mode = 'rgb', batch_size = 1, seed=15, shuffle=False)
21 ModelGenerators.append(ModelGenerator4)

(x)
Found 60 images belonging to 4 classes.

(y)
1 estimator = []
2 model1 = keras.models.load_model('/content/drive/MyDrive/218007620_HonorsProject/Model/best_' + name + 'B2.h5')
3 estimator.append(model1)
4 model2 = keras.models.load_model('/content/drive/MyDrive/218007620_HonorsProject/Model/best_' + name + 'B3.h5')
5 estimator.append(model2)
6 model3 = keras.models.load_model('/content/drive/MyDrive/218007620_HonorsProject/Model/best_' + name + 'B6.h5')
7 estimator.append(model3)
8 model4 = keras.models.load_model('/content/drive/MyDrive/218007620_HonorsProject/Model/best_' + name + 'B0.h5')
9 estimator.append(model4)
10
11 predictions = []
12 for i in range(len(estimator)):
13     prediction = estimator[i].predict(ModelGenerators[i], verbose = 1)
14     prediction = np.argmax(prediction, axis = 1)
15     predictions.append(prediction)
16
17
18
```

\*Please note the models take about 5-10 hours to train. It may require tedious amounts of time. Disconnections may result in re running the code. Please do not run if you do not have a UPS (due to loadshedding) as it may be frustrating to run and wait all over again.

Hope this helps and feel free to contact me via email – [nikielramawthar@gmail.com](mailto:nikielramawthar@gmail.com) if you are unclear about certain details pertaining to this.