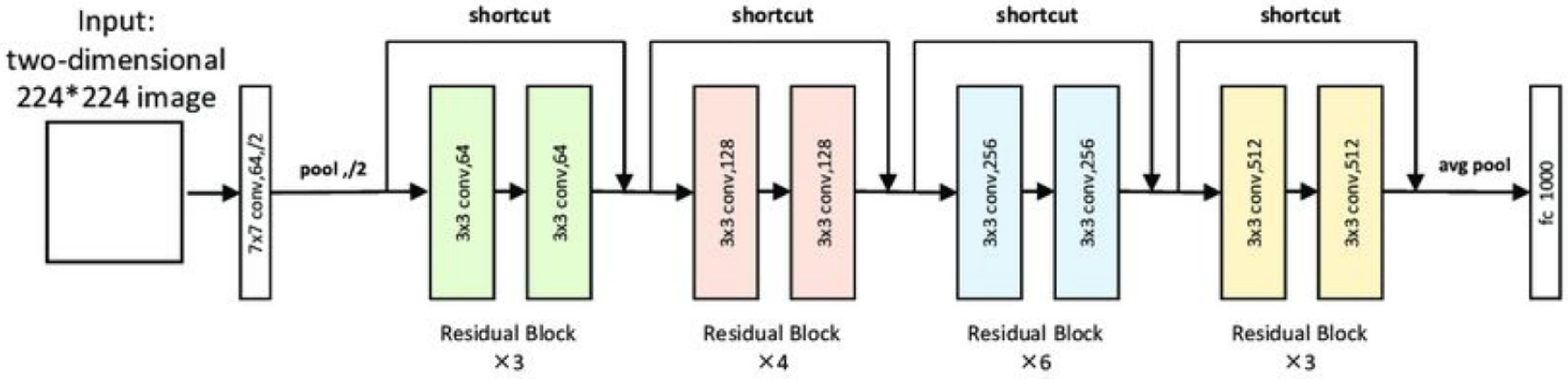
Final report

Member：

Date：12/18

AD Data

Model selection – ResNet34



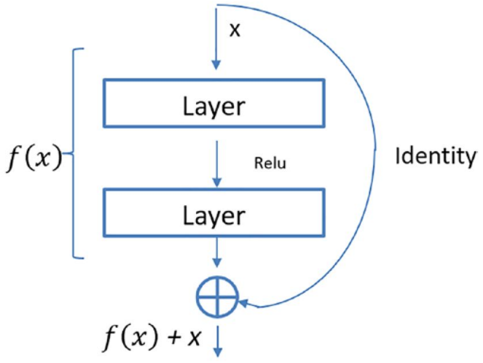
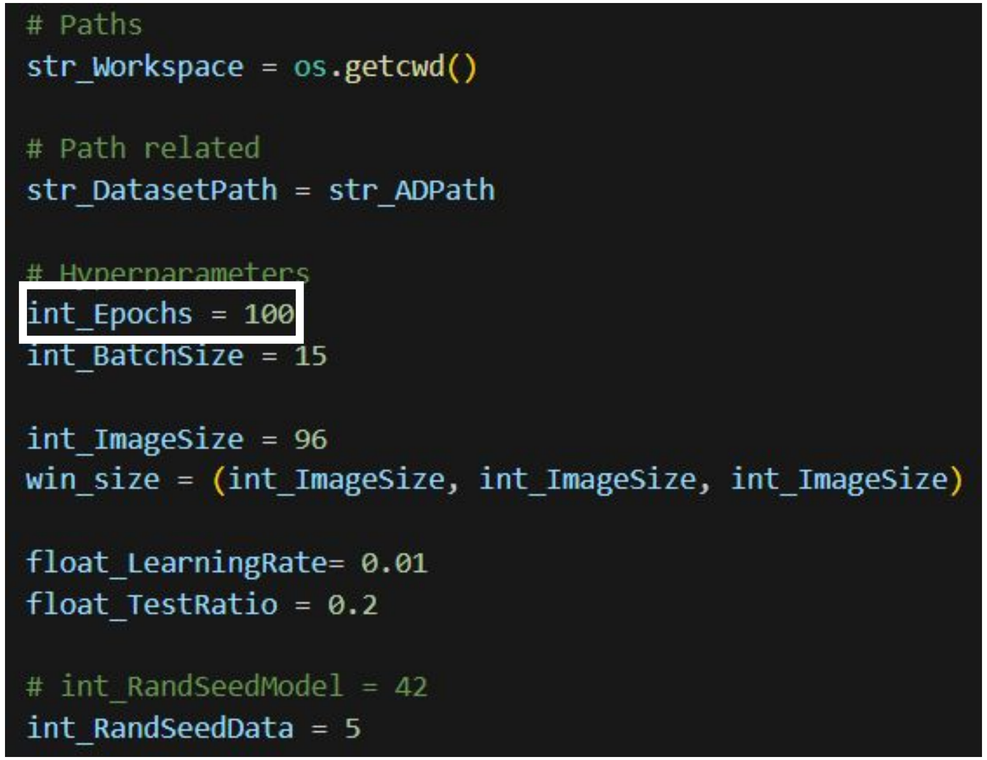


Figure 1-20. *Residual structures*

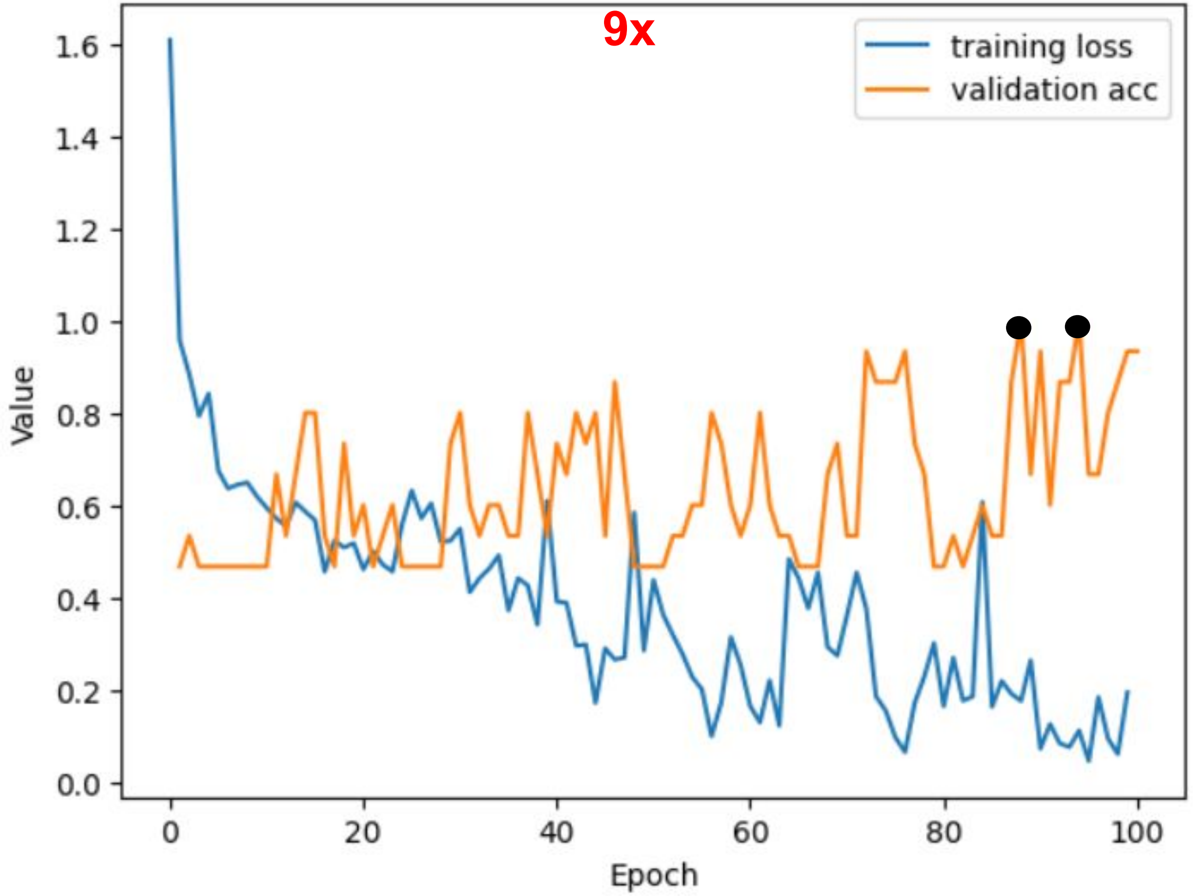
Solve gradient vanishing/exploding to improve model preformance

Training – selection best model

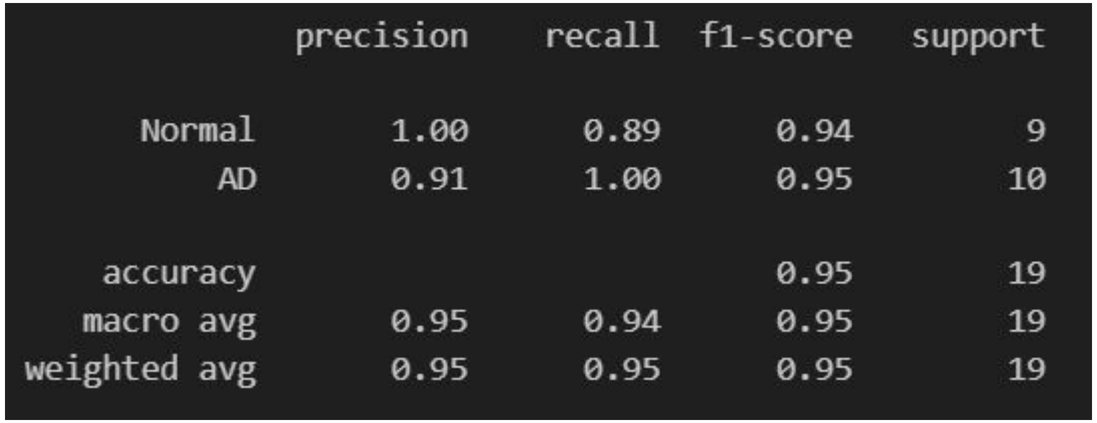


```
# Paths
str_Workspace = os.getcwd()

# Path related
str_DatasetPath = str_ADPath

# Hyperparameters
int_Epochs = 100
int_BatchSize = 15

int_ImageSize = 96
win_size = (int_ImageSize, int_ImageSize, int_ImageSize)

float_LearningRate= 0.01
float_TestRatio = 0.2

# int_RandSeedModel = 42
int_RandSeedData = 5
```

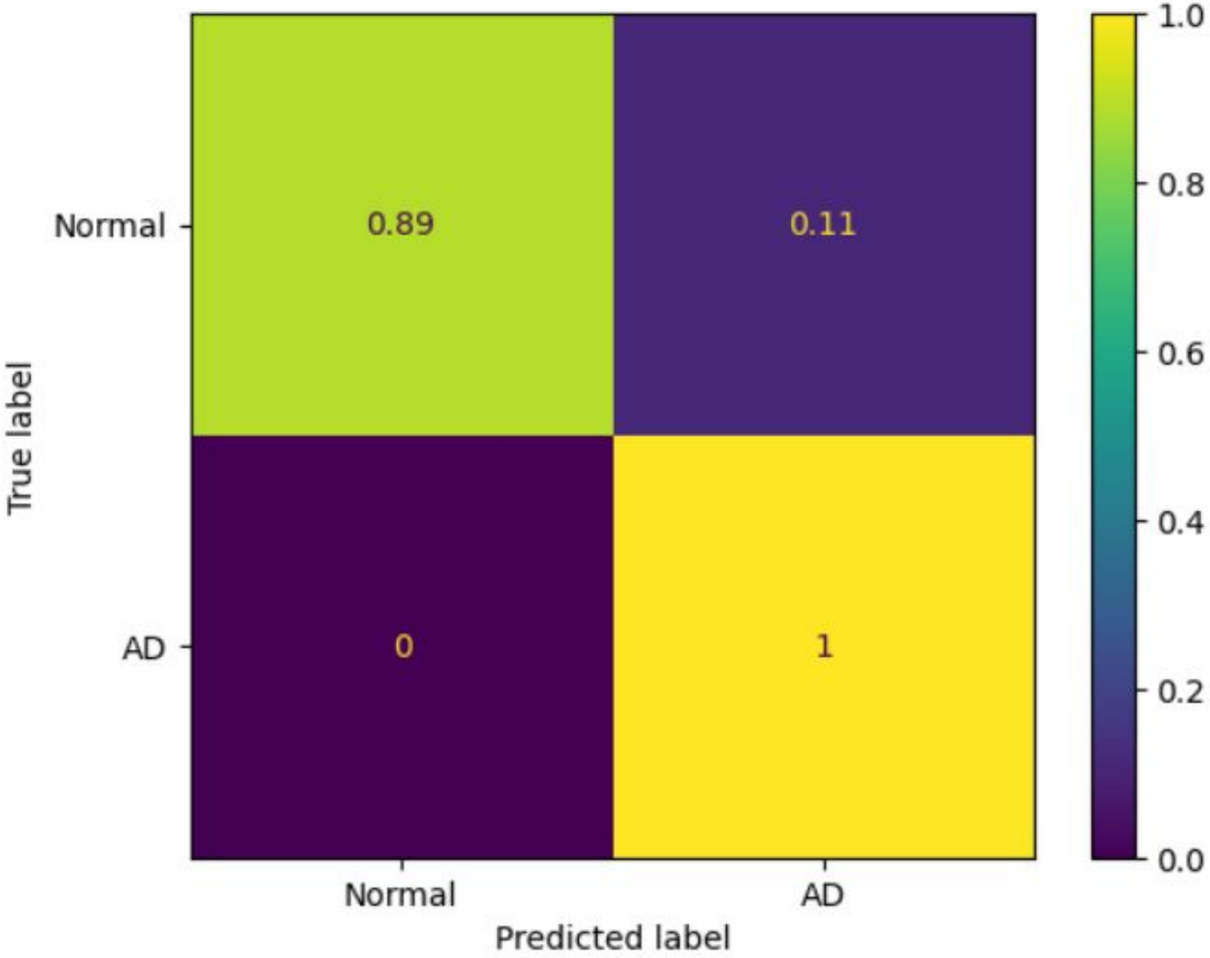
Validation data 100% accuracy in epoch 8x, 9x



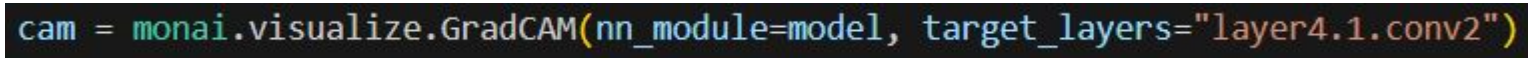
Evaluation metrics – test data



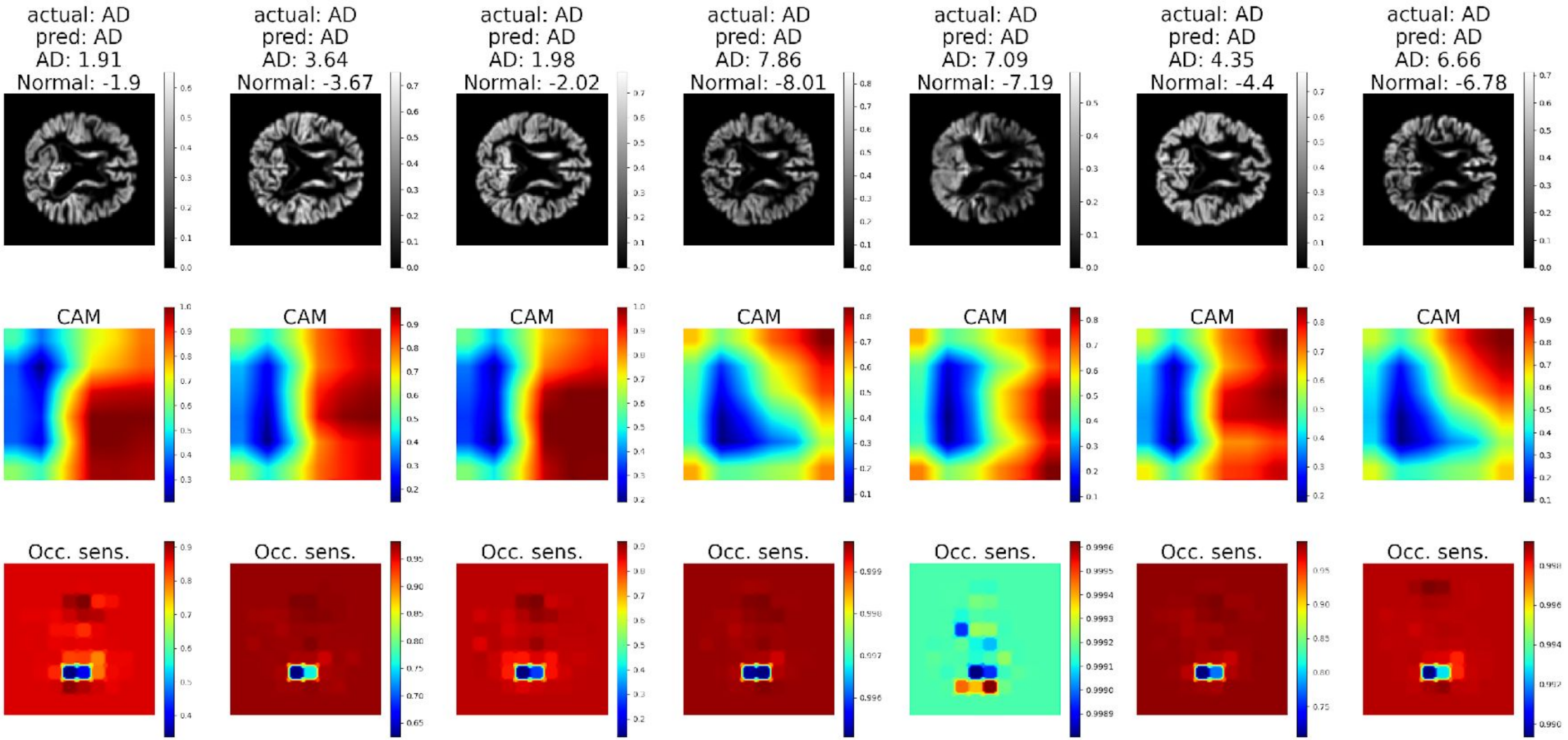
```
              precision    recall  f1-score   support

      Normal       1.00      0.89      0.94         9
          AD       0.91      1.00      0.95        10

    accuracy                           0.95        19
   macro avg       0.95      0.94      0.95        19
weighted avg       0.95      0.95      0.95        19
```



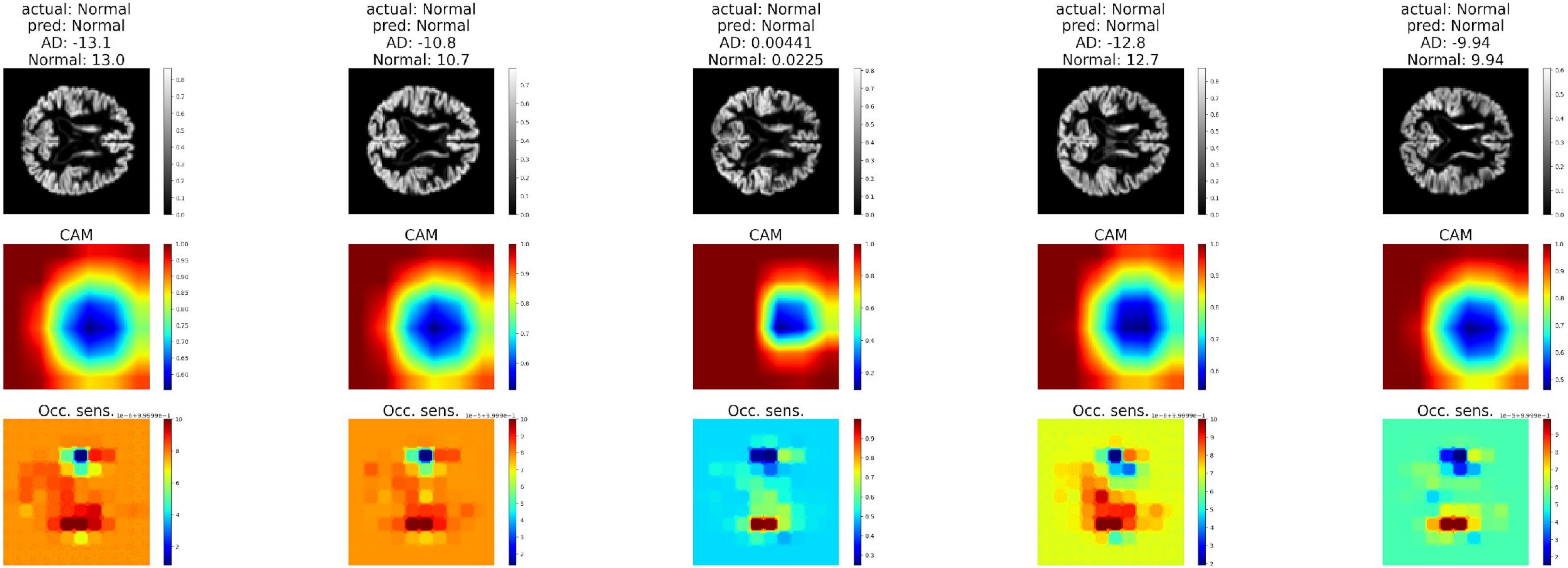
Interpretability - Patients



```
cam = monai.visualize.GradCAM(nn_module=model, target_layers="layer4.1.conv2")
```

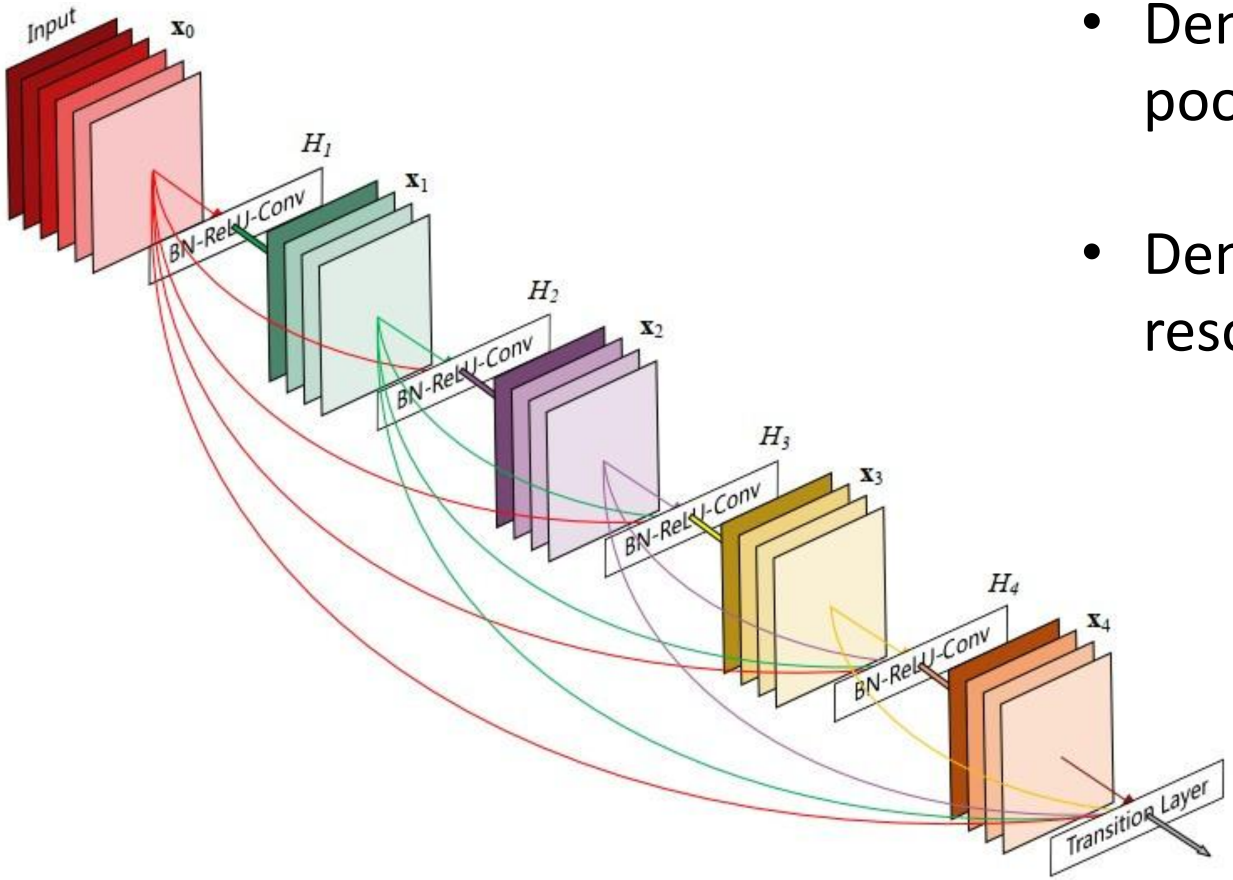


Interpretability - Normal person



ASD Data

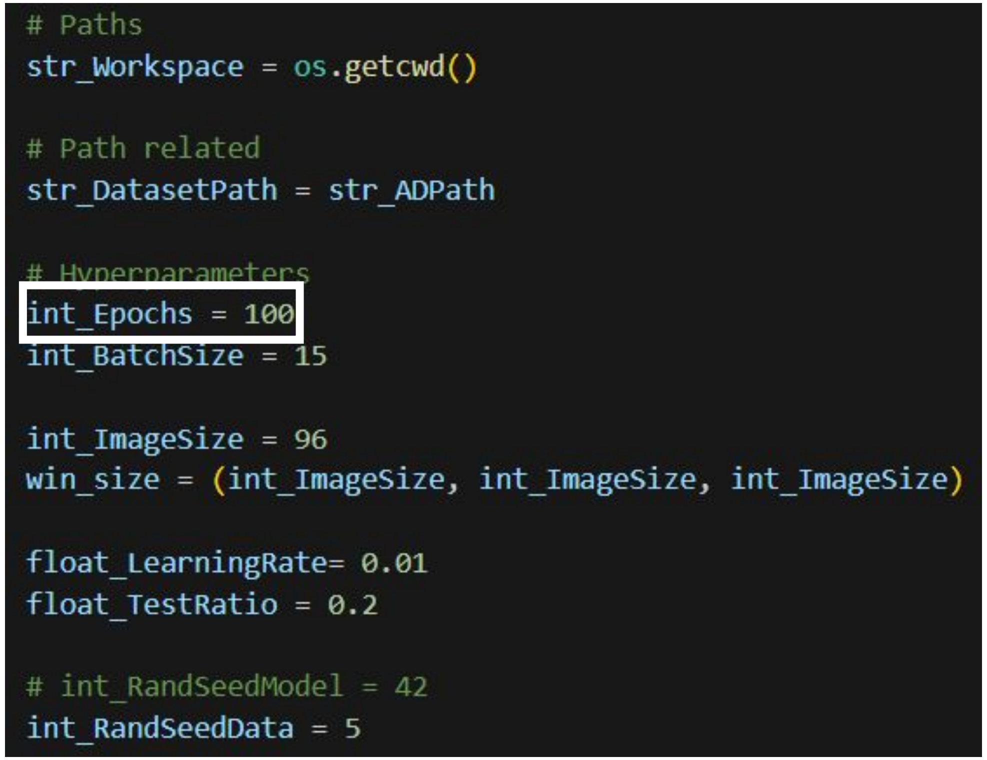
Model selection – DenseNet121



- DenseNet121 and ResNet34 demonstrate poor accuracy

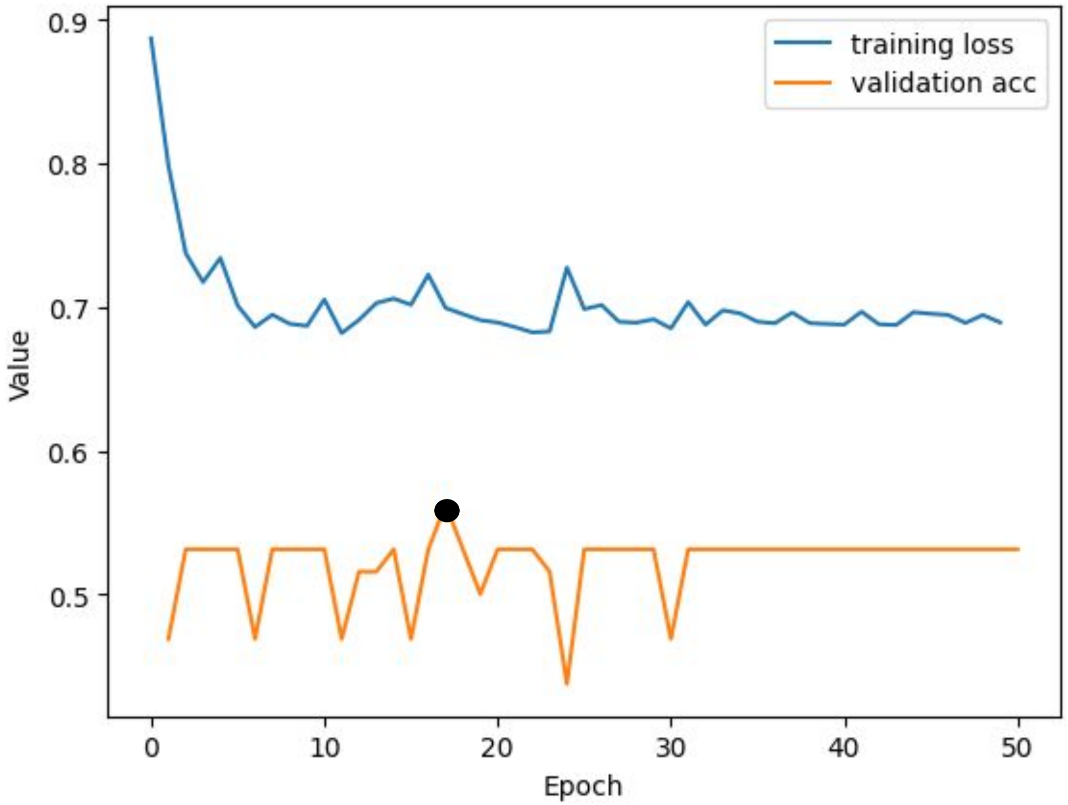
- DenseNet121 uses fewer computational resources

Training – selection best model

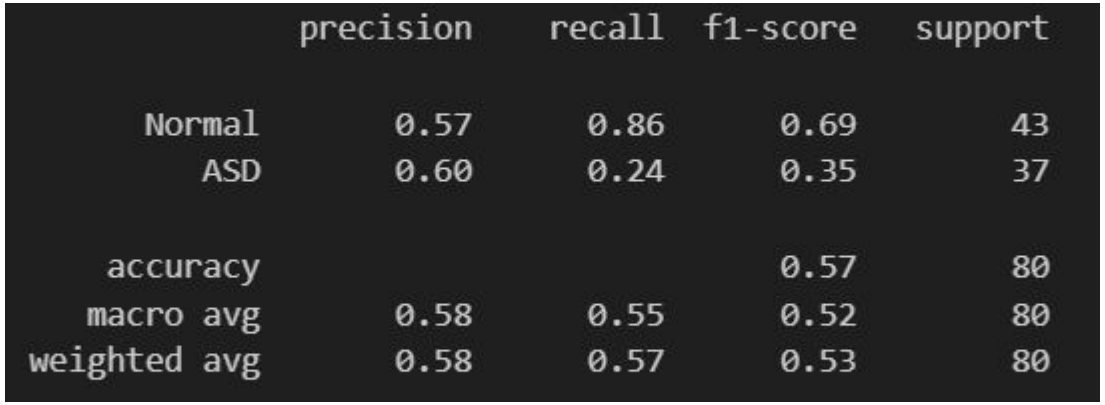


```
# Paths
str_Workspace = os.getcwd()

# Path related
str_DatasetPath = str_ADPath

# Hyperparameters
int_Epochs = 100
int_BatchSize = 15

int_ImageSize = 96
win_size = (int_ImageSize, int_ImageSize, int_ImageSize)

float_LearningRate= 0.01
float_TestRatio = 0.2

# int_RandSeedModel = 42
int_RandSeedData = 5
```

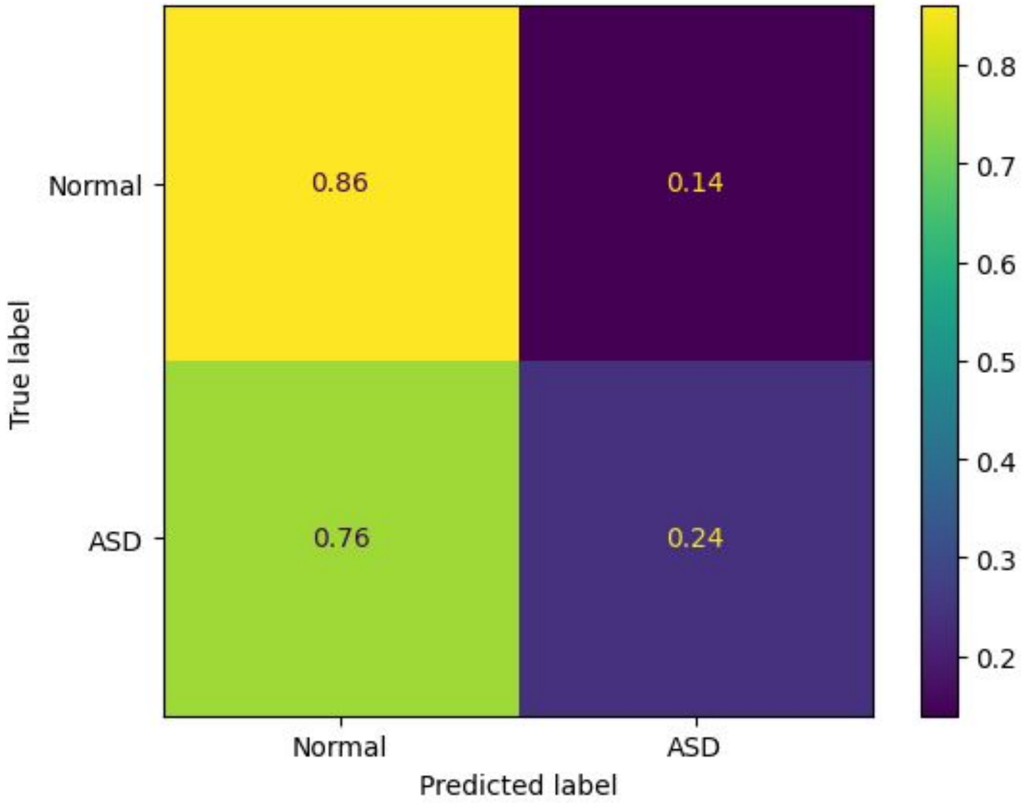
Validation data 56.25% accuracy in epoch 17



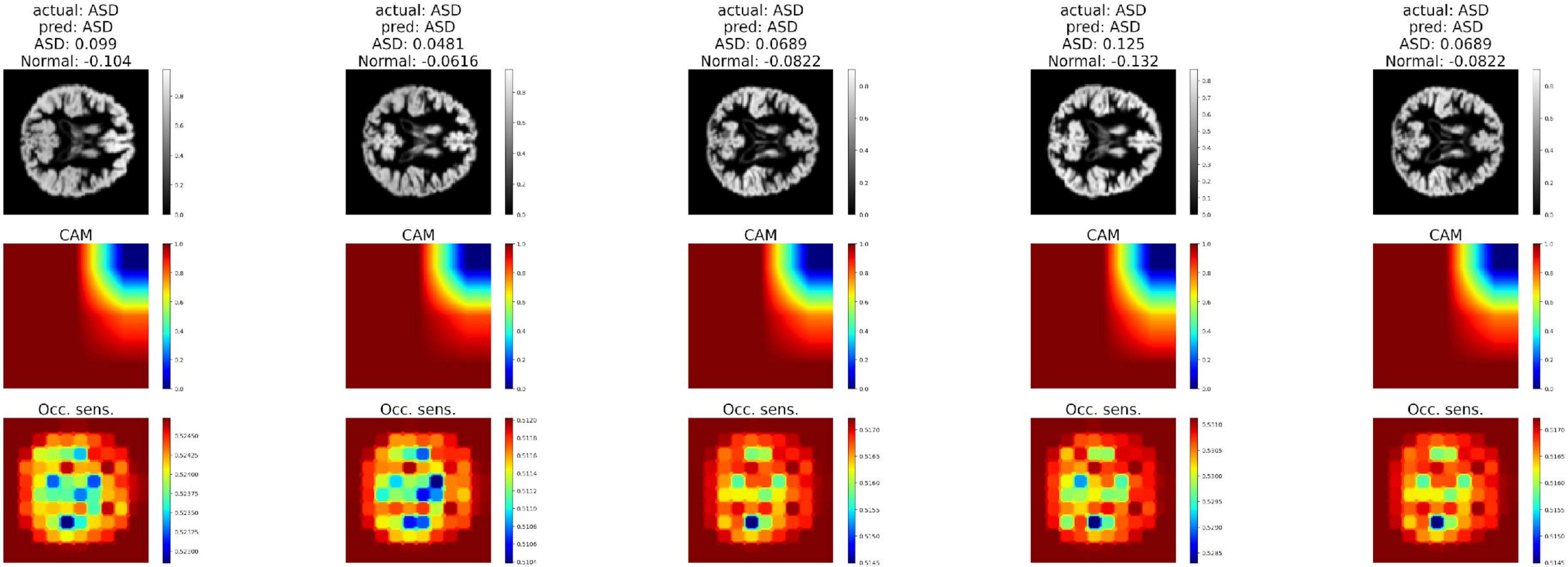
Evaluation metrics – test data



```
              precision    recall  f1-score   support

      Normal       0.57      0.86      0.69        43
         ASD       0.60      0.24      0.35        37

    accuracy                           0.57        80
   macro avg       0.58      0.55      0.52        80
weighted avg       0.58      0.57      0.53        80
```



Interpretability - Patients



actual: ASD
pred: ASD
ASD: 0.099
Normal: -0.104

actual: ASD
pred: ASD
ASD: 0.0481
Normal: -0.0616

actual: ASD
pred: ASD
ASD: 0.0689
Normal: -0.0822

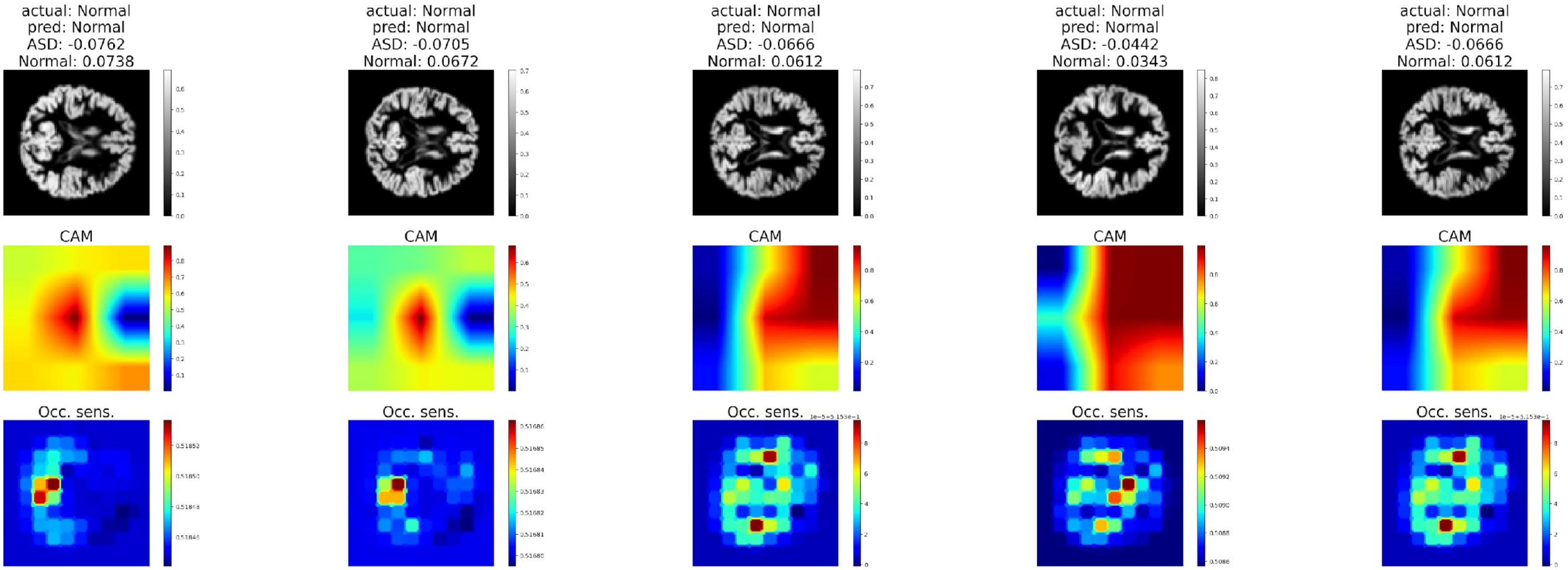
actual: ASD
pred: ASD
ASD: 0.125
Normal: -0.132

actual: ASD
pred: ASD
ASD: 0.0689
Normal: -0.0822

CAM

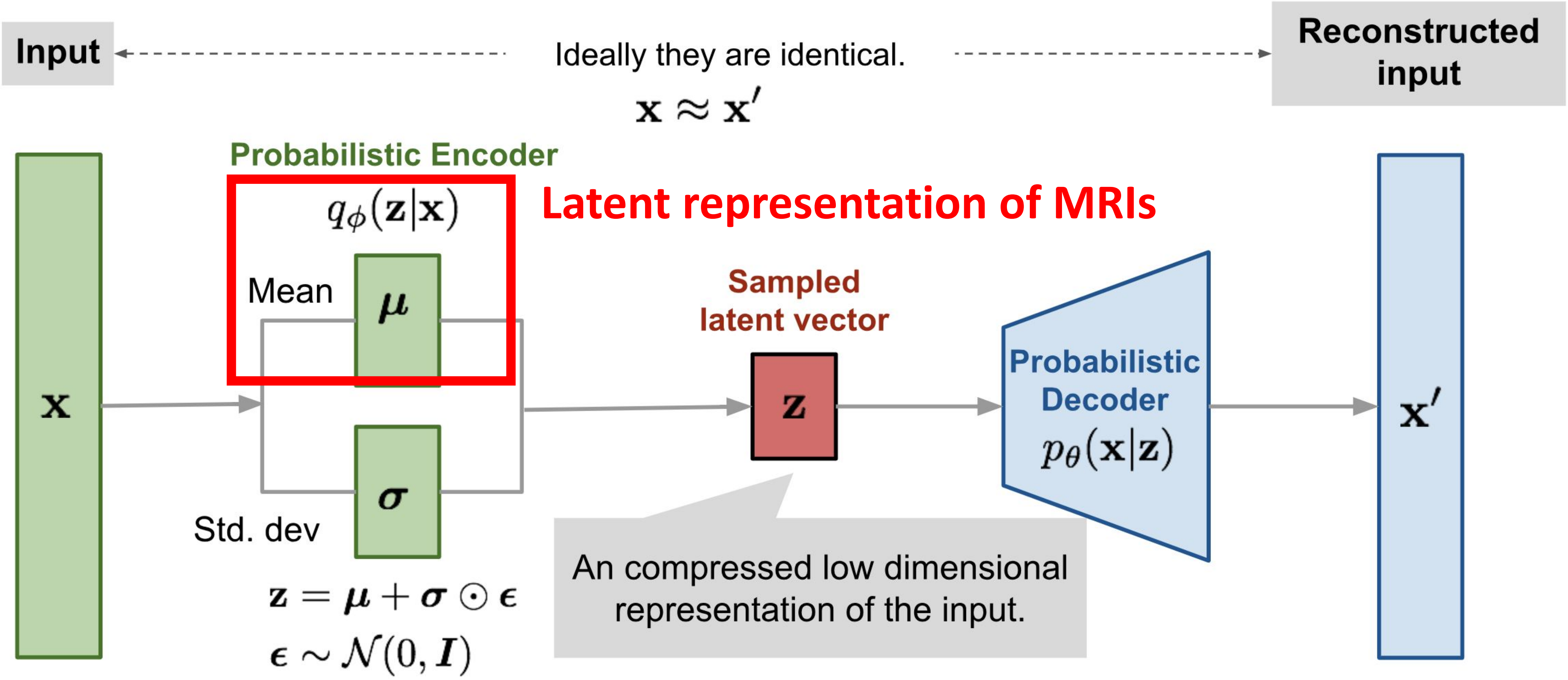
Occ. sens.

Interpretability - Normal person



Bonus –
three-class classification
or multimodel

Model selection – Variational autoencoder



Input

Ideally they are identical.

$\mathbf{x} \approx \mathbf{x}'$

Reconstructed input

Probabilistic Encoder

$q_\phi(\mathbf{z}|\mathbf{x})$

Latent representation of MRIs

$\mathbf{x}$

Mean $\boldsymbol{\mu}$

Std. dev $\boldsymbol{\sigma}$

$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$

$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$

Sampled latent vector

$\mathbf{z}$

An compressed low dimensional representation of the input.

Probabilistic Decoder

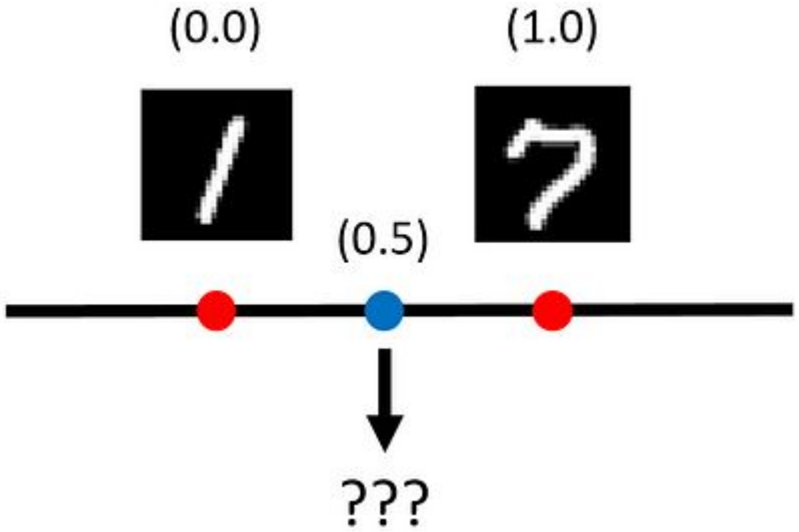
$p_\theta(\mathbf{x}|\mathbf{z})$

$\mathbf{x}'$

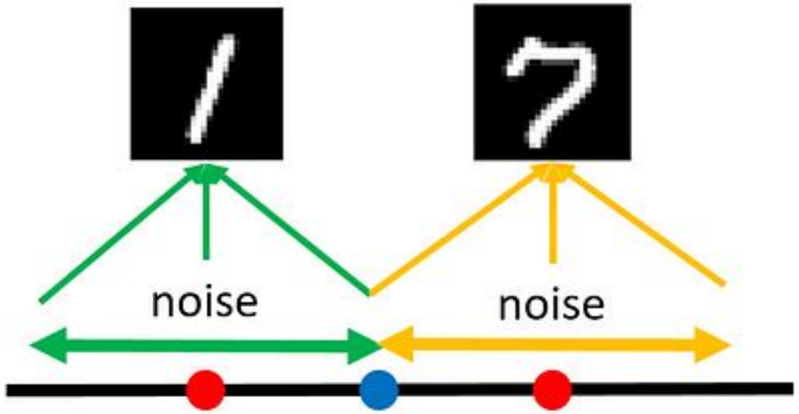
$$Loss = \sum x \log x' + (1-x)\log(1-x') - \frac{1}{2}\sum(1 + \log(\sigma_i{}^2) - m_i{}^2 - \sigma_i{}^2)$$

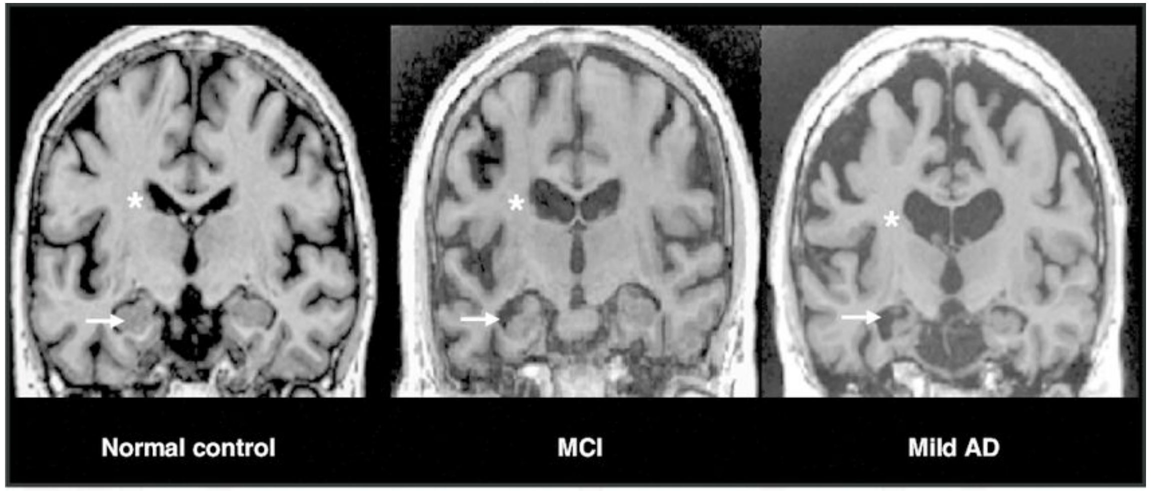
Model selection – Variational autoencoder

AE:



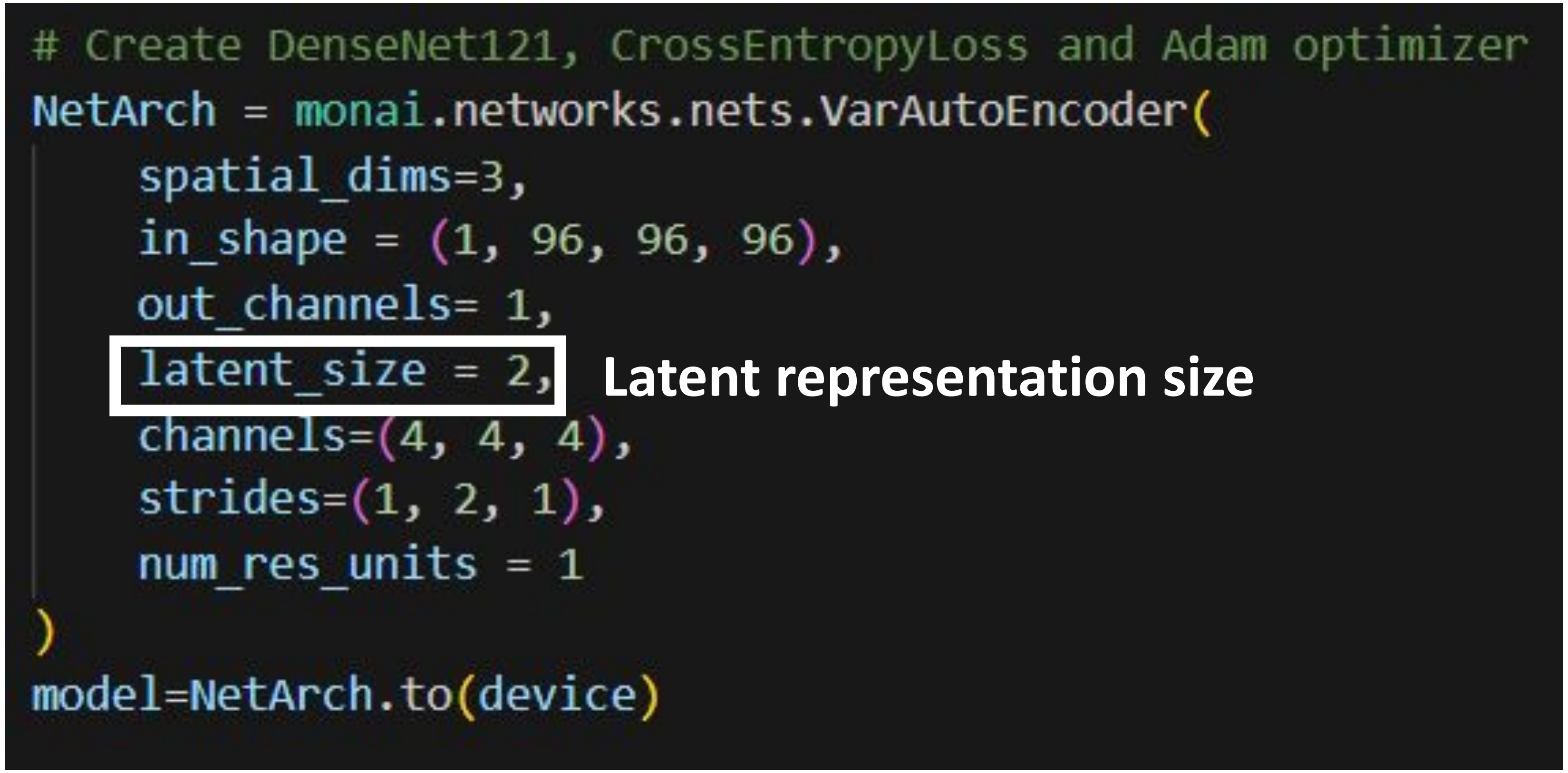
VAE:





Learning progressive disease
to extract latent representation

Model architecture

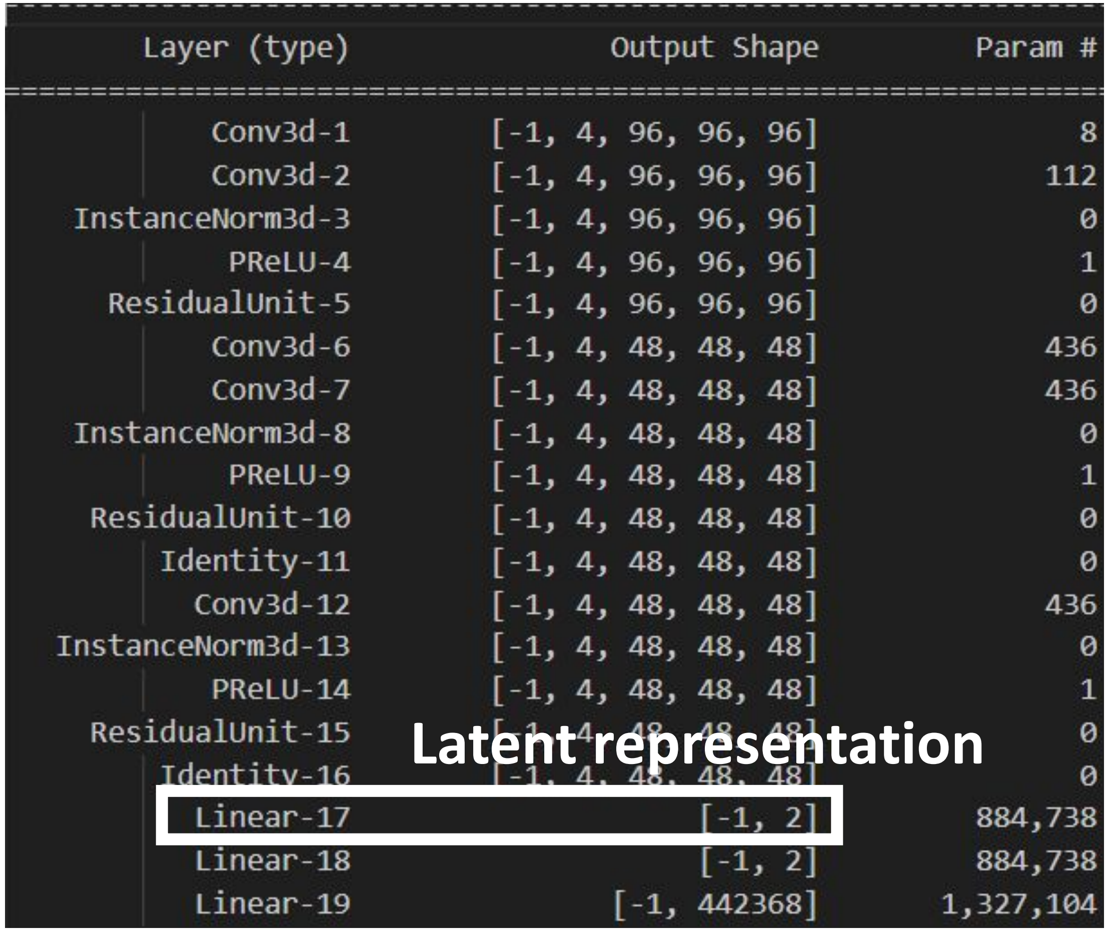


```
# Create DenseNet121, CrossEntropyLoss and Adam optimizer
NetArch = monai.networks.nets.VarAutoEncoder(
    spatial_dims=3,
    in_shape = (1, 96, 96, 96),
    out_channels= 1,
    latent_size = 2,
    channels=(4, 4, 4),
    strides=(1, 2, 1),
    num_res_units = 1
)
model=NetArch.to(device)
```

Latent representation size

Model architecture

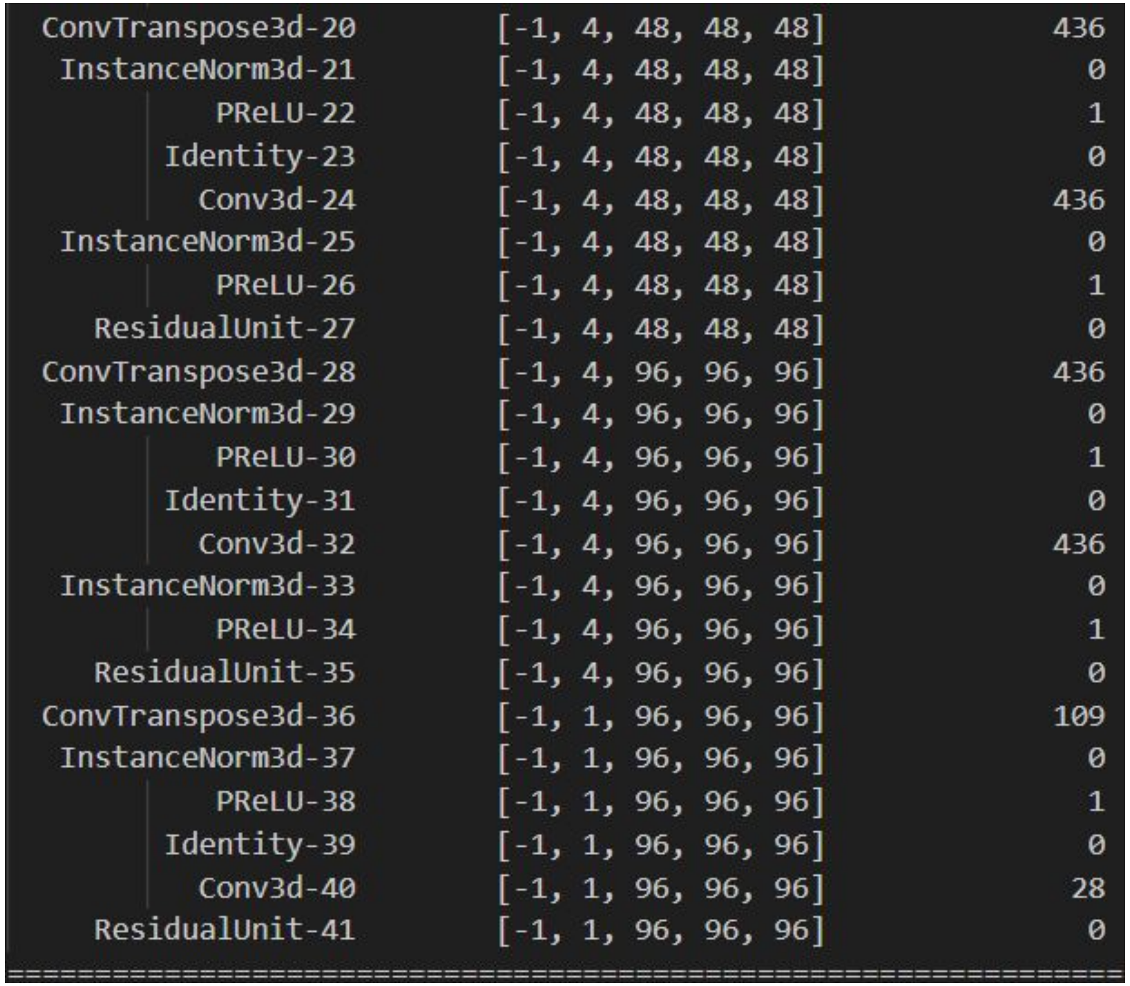
Encoder：



```
================================================================
        Layer (type)               Output Shape         Param #
================================================================
            Conv3d-1       [-1, 4, 96, 96, 96]               8
            Conv3d-2       [-1, 4, 96, 96, 96]             112
    InstanceNorm3d-3       [-1, 4, 96, 96, 96]               0
             PReLU-4       [-1, 4, 96, 96, 96]               1
      ResidualUnit-5       [-1, 4, 96, 96, 96]               0
            Conv3d-6       [-1, 4, 48, 48, 48]             436
            Conv3d-7       [-1, 4, 48, 48, 48]             436
    InstanceNorm3d-8       [-1, 4, 48, 48, 48]               0
             PReLU-9       [-1, 4, 48, 48, 48]               1
     ResidualUnit-10       [-1, 4, 48, 48, 48]               0
         Identity-11       [-1, 4, 48, 48, 48]               0
           Conv3d-12       [-1, 4, 48, 48, 48]             436
   InstanceNorm3d-13       [-1, 4, 48, 48, 48]               0
            PReLU-14       [-1, 4, 48, 48, 48]               1
     ResidualUnit-15       [-1, 4, 48, 48, 48]               0
         Identity-16       [-1, 4, 48, 48, 48]               0
           Linear-17                   [-1, 2]         884,738
           Linear-18                   [-1, 2]         884,738
           Linear-19              [-1, 442368]       1,327,104
```

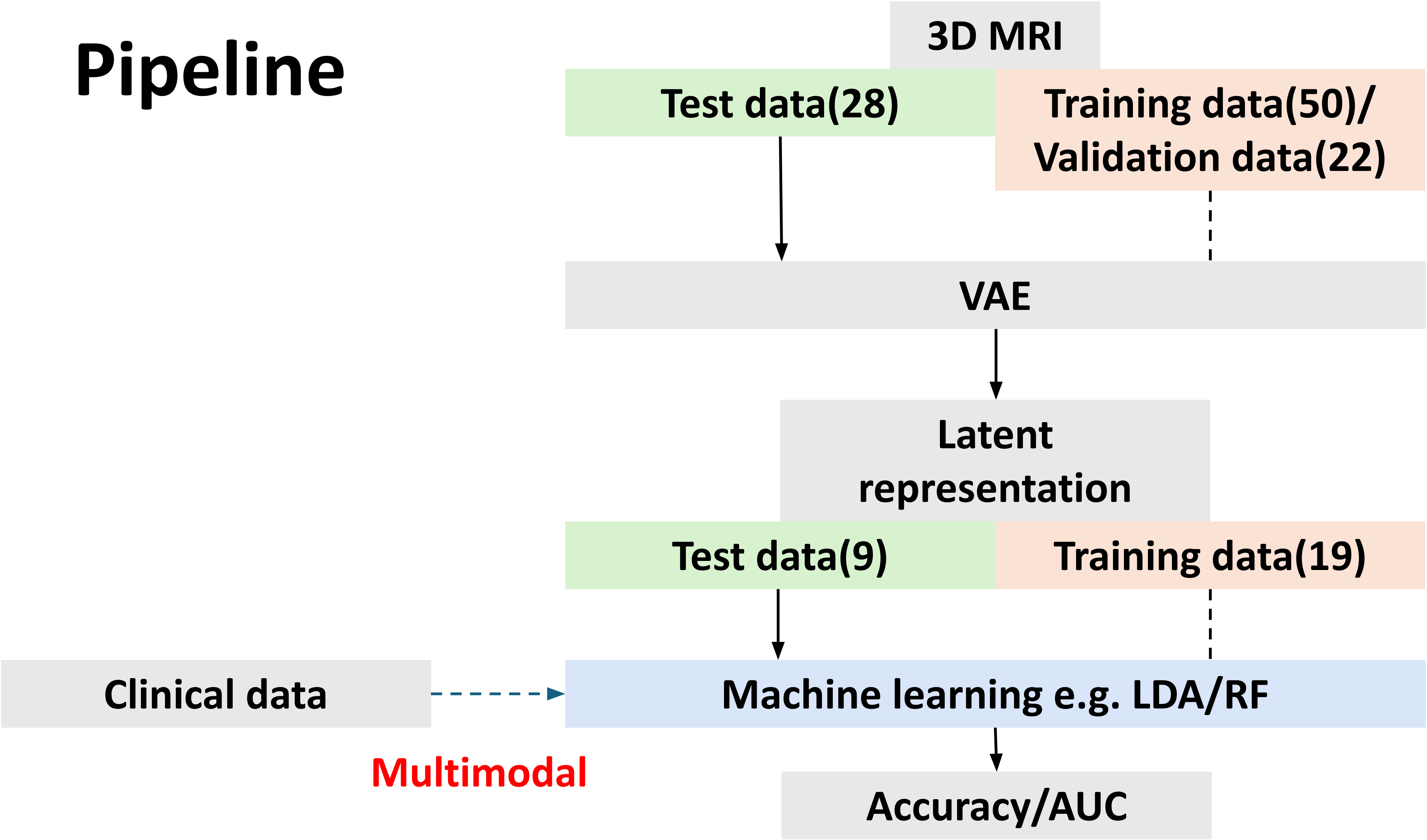
Latent representation

Decoder：



```
  ConvTranspose3d-20       [-1, 4, 48, 48, 48]             436
   InstanceNorm3d-21       [-1, 4, 48, 48, 48]               0
            PReLU-22       [-1, 4, 48, 48, 48]               1
         Identity-23       [-1, 4, 48, 48, 48]               0
           Conv3d-24       [-1, 4, 48, 48, 48]             436
   InstanceNorm3d-25       [-1, 4, 48, 48, 48]               0
            PReLU-26       [-1, 4, 48, 48, 48]               1
     ResidualUnit-27       [-1, 4, 48, 48, 48]               0
  ConvTranspose3d-28       [-1, 4, 96, 96, 96]             436
   InstanceNorm3d-29       [-1, 4, 96, 96, 96]               0
            PReLU-30       [-1, 4, 96, 96, 96]               1
         Identity-31       [-1, 4, 96, 96, 96]               0
           Conv3d-32       [-1, 4, 96, 96, 96]             436
   InstanceNorm3d-33       [-1, 4, 96, 96, 96]               0
            PReLU-34       [-1, 4, 96, 96, 96]               1
     ResidualUnit-35       [-1, 4, 96, 96, 96]               0
  ConvTranspose3d-36       [-1, 1, 96, 96, 96]             109
   InstanceNorm3d-37       [-1, 1, 96, 96, 96]               0
            PReLU-38       [-1, 1, 96, 96, 96]               1
         Identity-39       [-1, 1, 96, 96, 96]               0
           Conv3d-40       [-1, 1, 96, 96, 96]              28
     ResidualUnit-41       [-1, 1, 96, 96, 96]               0
================================================================
```

Pipeline



3D MRI

Test data(28)

Training data(50)/
Validation data(22)

VAE

Latent representation

Test data(9)

Training data(19)

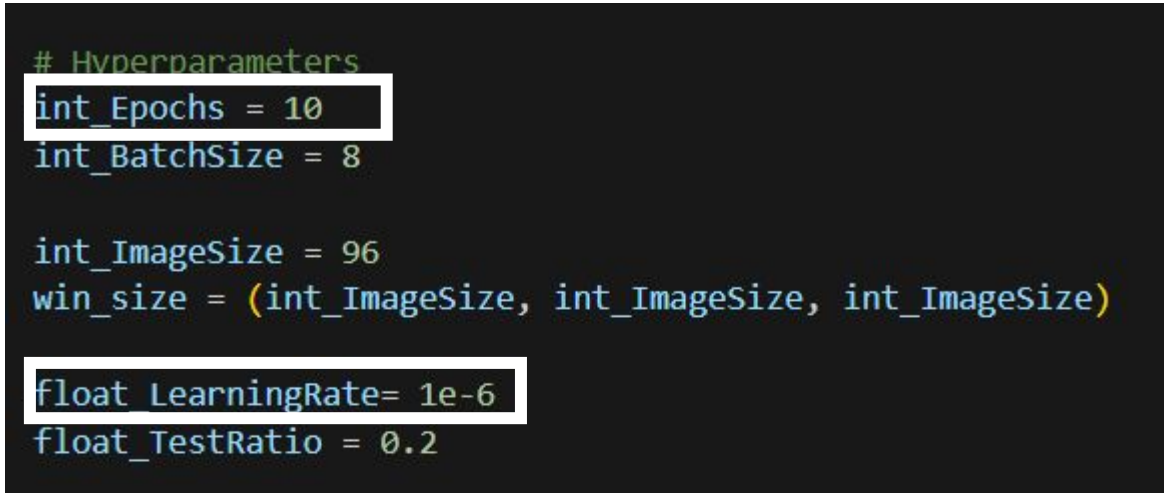
Clinical data

Machine learning e.g. LDA/RF

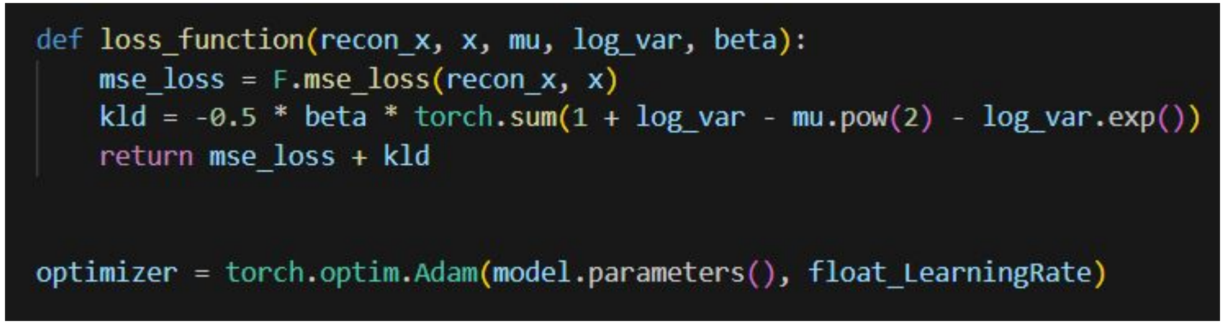
Multimodal

Accuracy/AUC

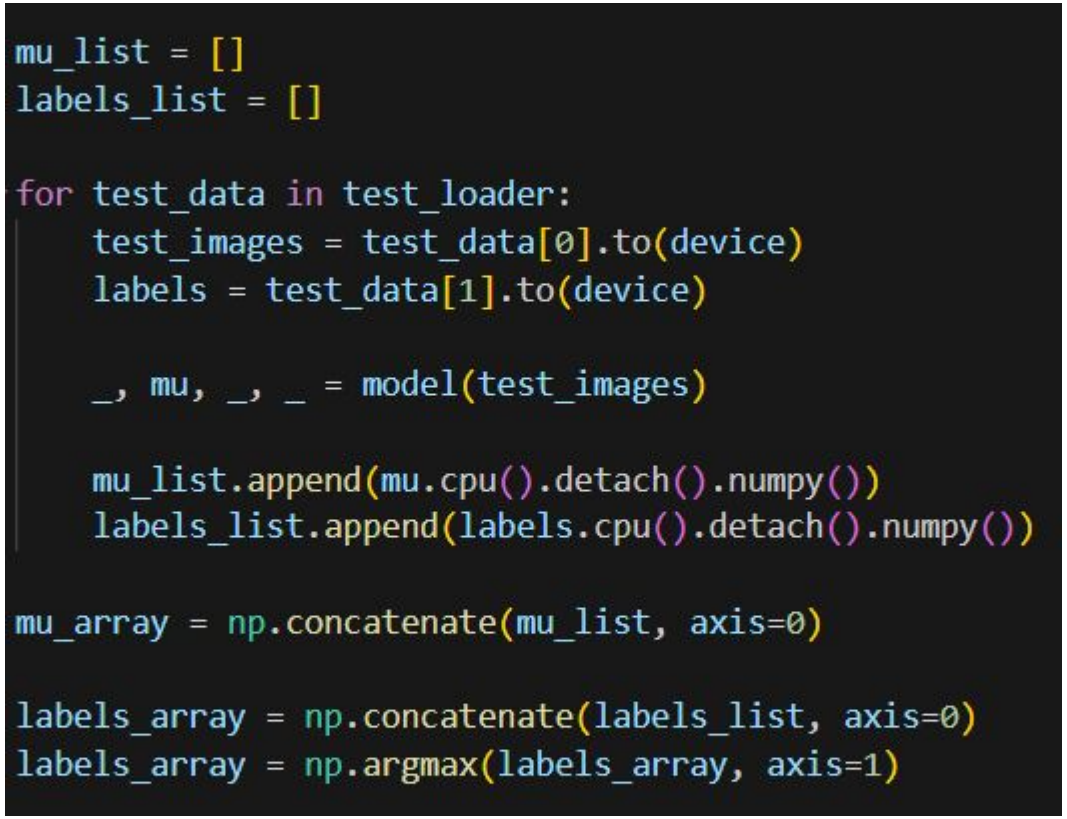
Training model



```
# Hyperparameters
int_Epochs = 10
int_BatchSize = 8

int_ImageSize = 96
win_size = (int_ImageSize, int_ImageSize, int_ImageSize)

float_LearningRate= 1e-6
float_TestRatio = 0.2
```

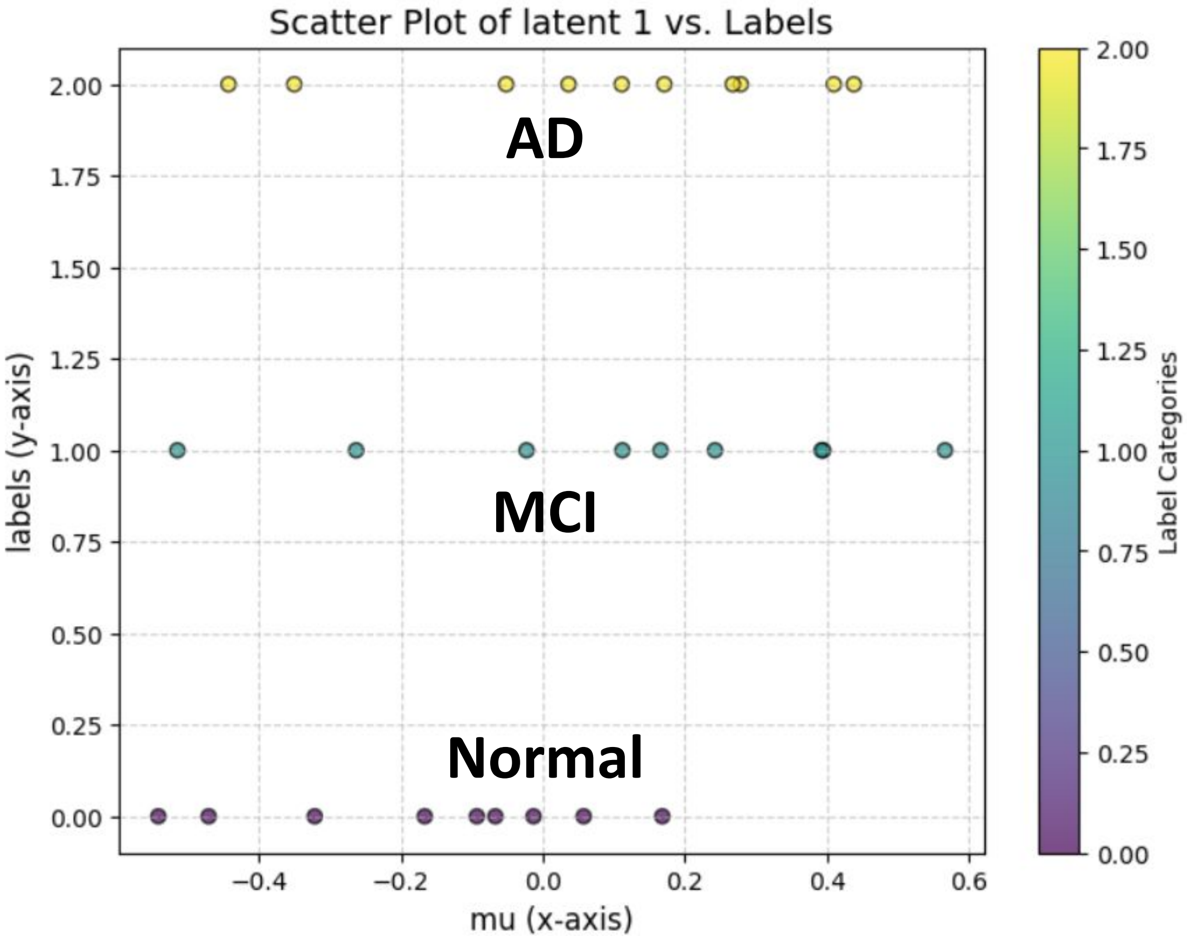


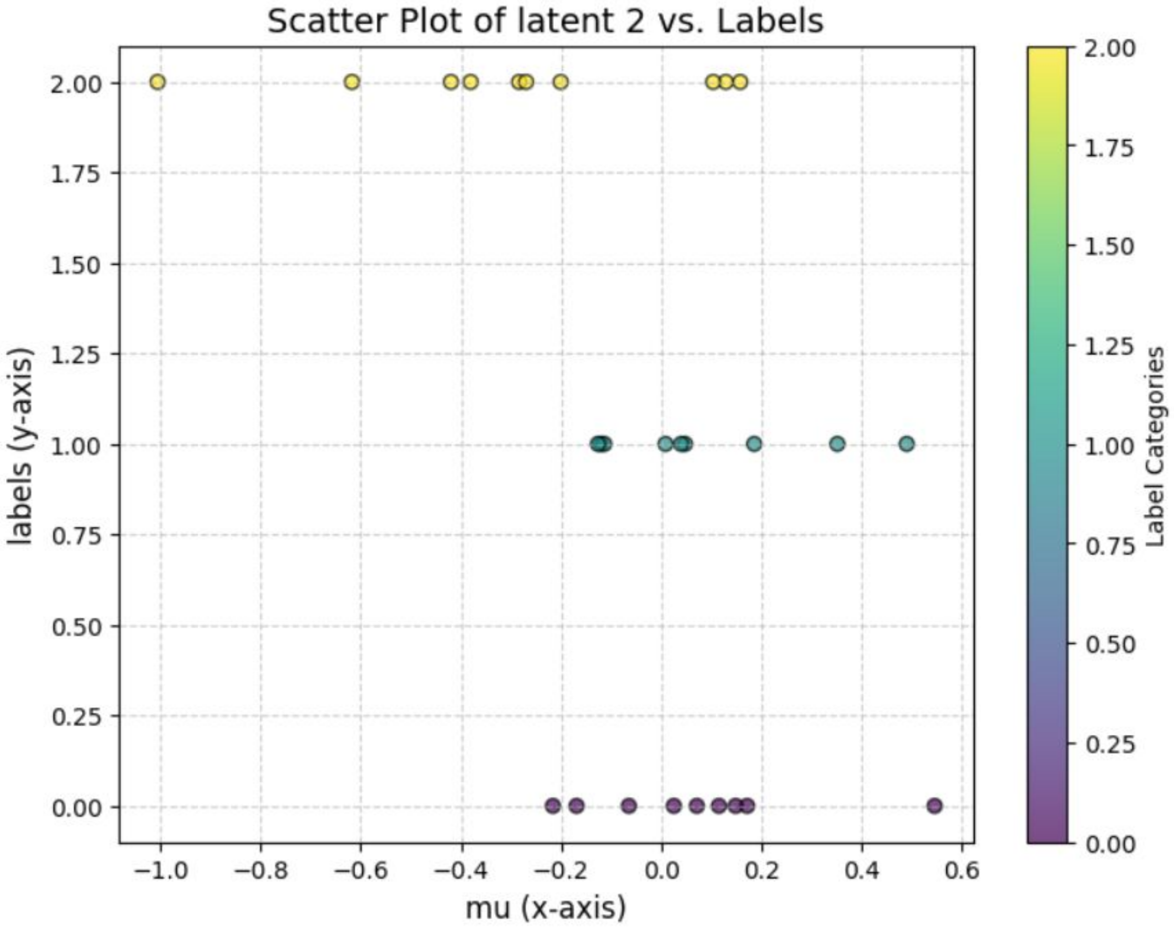
```
def loss_function(recon_x, x, mu, log_var, beta):
    mse_loss = F.mse_loss(recon_x, x)
    kld = -0.5 * beta * torch.sum(1 + log_var - mu.pow(2) - log_var.exp())
    return mse_loss + kld


optimizer = torch.optim.Adam(model.parameters(), float_LearningRate)
```



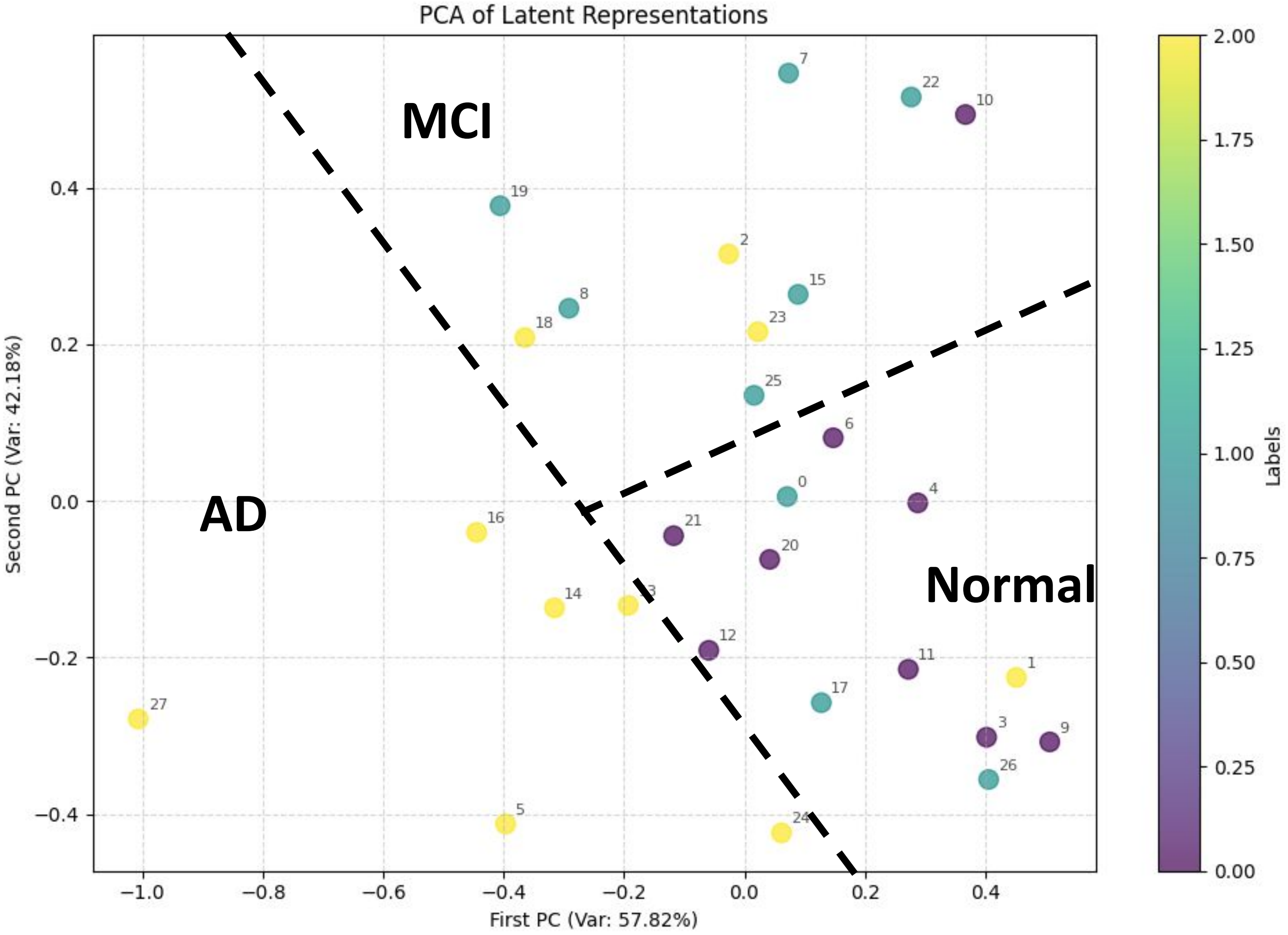
```
mu_list = []
labels_list = []

for test_data in test_loader:
    test_images = test_data[0].to(device)
    labels = test_data[1].to(device)

    _, mu, _, _ = model(test_images)

    mu_list.append(mu.cpu().detach().numpy())
    labels_list.append(labels.cpu().detach().numpy())

mu_array = np.concatenate(mu_list, axis=0)

labels_array = np.concatenate(labels_list, axis=0)
labels_array = np.argmax(labels_array, axis=1)
```

Analyze latent representations

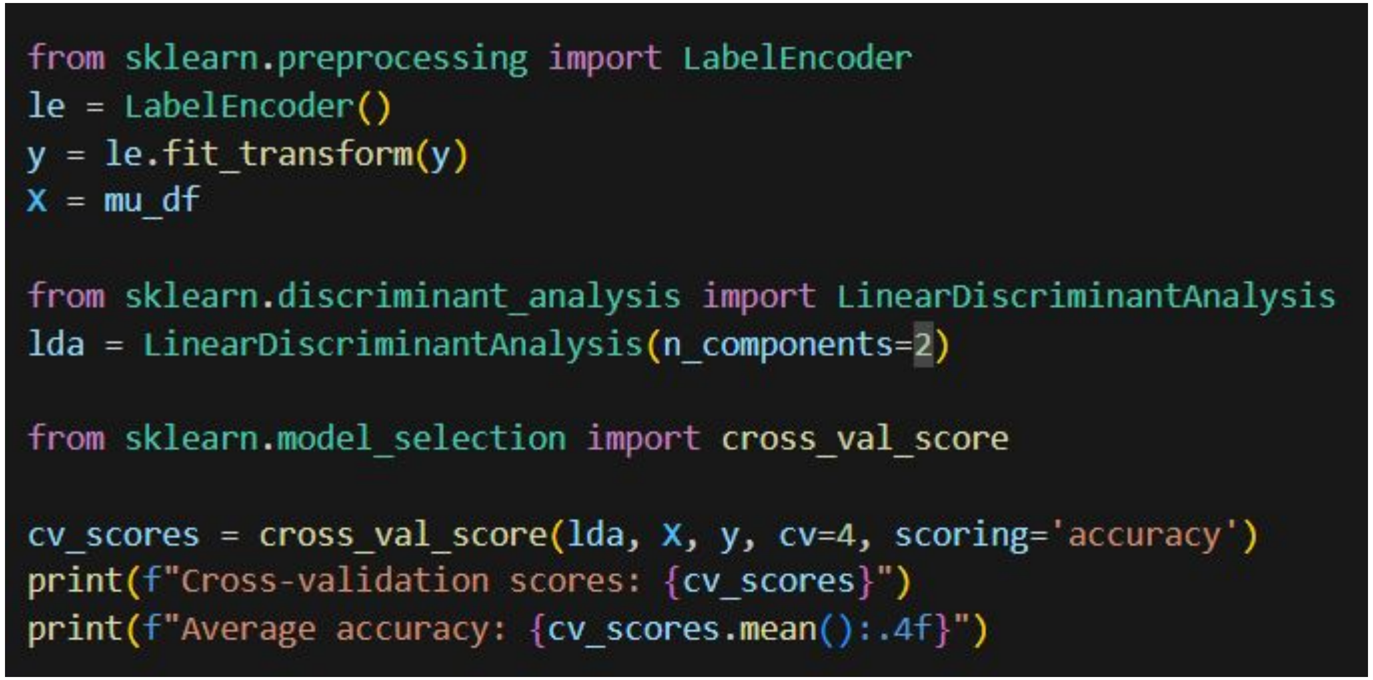




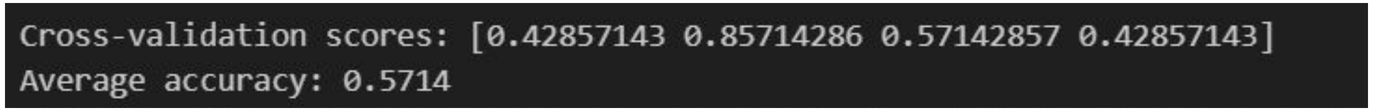
Embeddings(mu) - PCA



Machine learning performance (LDA)

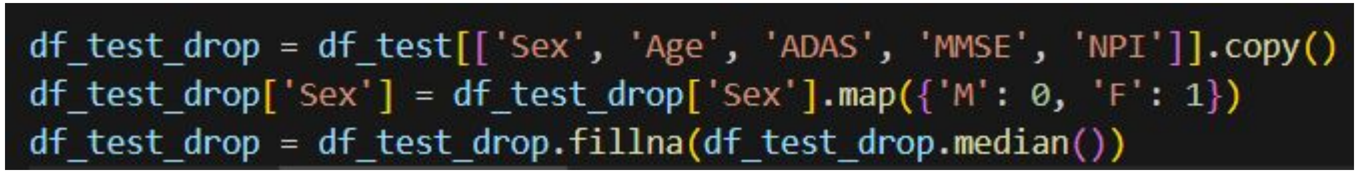


```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)
X = mu_df

from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
lda = LinearDiscriminantAnalysis(n_components=2)

from sklearn.model_selection import cross_val_score

cv_scores = cross_val_score(lda, X, y, cv=4, scoring='accuracy')
print(f"Cross-validation scores: {cv_scores}")
print(f"Average accuracy: {cv_scores.mean():.4f}")
```

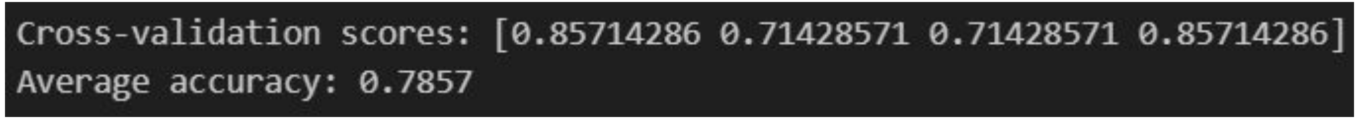


```
Cross-validation scores: [0.42857143 0.85714286 0.57142857 0.42857143]
Average accuracy: 0.5714
```

Multimodal (LDA)

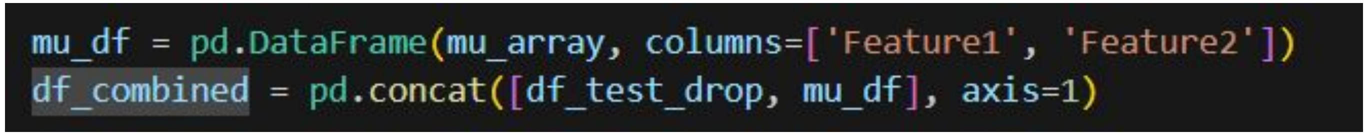


```
df_test_drop = df_test[['Sex', 'Age', 'ADAS', 'MMSE', 'NPI']].copy()
df_test_drop['Sex'] = df_test_drop['Sex'].map({'M': 0, 'F': 1})
df_test_drop = df_test_drop.fillna(df_test_drop.median())
```

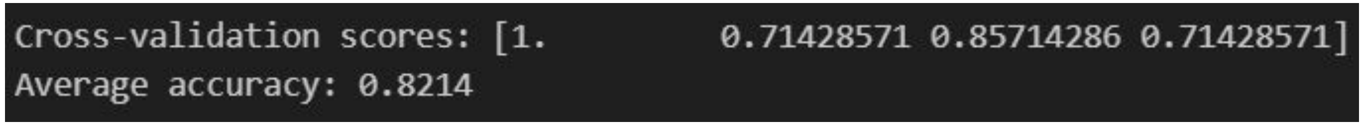


```
Cross-validation scores: [0.85714286 0.71428571 0.71428571 0.85714286]
Average accuracy: 0.7857
```

Concat latent representation



```
mu_df = pd.DataFrame(mu_array, columns=['Feature1', 'Feature2'])
df_combined = pd.concat([df_test_drop, mu_df], axis=1)
```



```
Cross-validation scores: [1.         0.71428571 0.85714286 0.71428571]
Average accuracy: 0.8214
```