

- a. How did you address the problems of imbalanced data and insufficient sample size? (5 pts)

用 RandomOverSampler 這個函式庫

Random Oversampling

```
[100] ✓ 0.0s
from imblearn.over_sampling import RandomOverSampler

oversampler = RandomOverSampler(random_state = 42)
train_data_ros_X, train_data_ros_y = oversampler.fit_resample(train_data.iloc[:, 1:5], train_data.iloc[:, 5])
train_data_ros = pd.concat([train_data_ros_X, train_data_ros_y], axis = 1)

[101] ✓ 0.0s

train_data_ros["Stage_New"].value_counts().sort_index()

[102] ✓ 0.0s
... Stage_New
0    55
1    55
2    55
3    55
4    55
5    55
Name: count, dtype: int64
```

- b. Reproducibility of the results (the first part 2 pts and the second part 4 pts)
基本上結果一樣
- c. Number of parameters: Please write the parameter count of your final selected model to the Kaggle competition (2 pts)

```
model = ResNet50plus(num_classes = 6)
total_params = sum(p.numel() for p in model.parameters())
print("總參數量:", total_params)
del model

✓ 0.3s

總參數量: 24035142
```

24035142

- d. The difficulty during training (the first part 3 pts and the second part 8 pts)
基本上跟上次一樣，就差在多了解決 imbalanced data
- e. Briefly explain the structures of the models you are using for the second part:
You are required to do analyses of single slice, late fusion, early fusion, and 3D CNNs (16 pts)

```

class ResNet50plus(nn.Module):
    def __init__(self, num_classes, input_size=(3, 50, 50), features_grad=False):
        super().__init__()
        # 本次作業指定使用Resnet50進行修改
        resnet50 = models.resnet50(weights = "IMAGENET1K_V1", progress = True)
        # Freeze the convolutional layers if features_grad is False
        if not features_grad:
            for param in resnet50.parameters():
                param.requires_grad = False
        # Replace the final fully connected layer with a custom one
        # ResNet50's final layer (fc) output is typically 2048-dimensional
        in_features = resnet50.fc.in_features
        # Modify the fully connected layer for custom input size and additional features
        self.resnet = nn.Sequential(
            *list(resnet50.children())[:-1], # Take all layers except the final fully connected layer
            nn.Flatten(),
        )
        # New fully connected layers to combine with additional inputs (e.g., age, gender)
        self.fc = nn.Sequential(
            nn.Linear(in_features + 2, 256), # +2 to account for age and gender features
            nn.BatchNorm1d(256),
            nn.ReLU(),
            nn.Dropout(0.5),
            nn.Linear(256, num_classes)
        )
    def forward(self, images, ages, genders):
        # Pass the image through ResNet backbone
        x = self.resnet(images)
        # Concatenate additional features (ages, genders) with the ResNet output
        x = torch.cat([x, ages.unsqueeze(1), genders.unsqueeze(1)], dim=1)
        # Pass concatenated features through fully connected layers
        x = self.fc(x)
        return x

```

基本上跟上次一樣，差別在因為這次輸出有 6 個，所以改了 self.fc 的部分

- f. You should submit compiled HTML file and ipynb notebook with name