KUBIG

CNN and MLP

실전 2반

조민제, 김도윤, 이지현, 이노아, 오석준

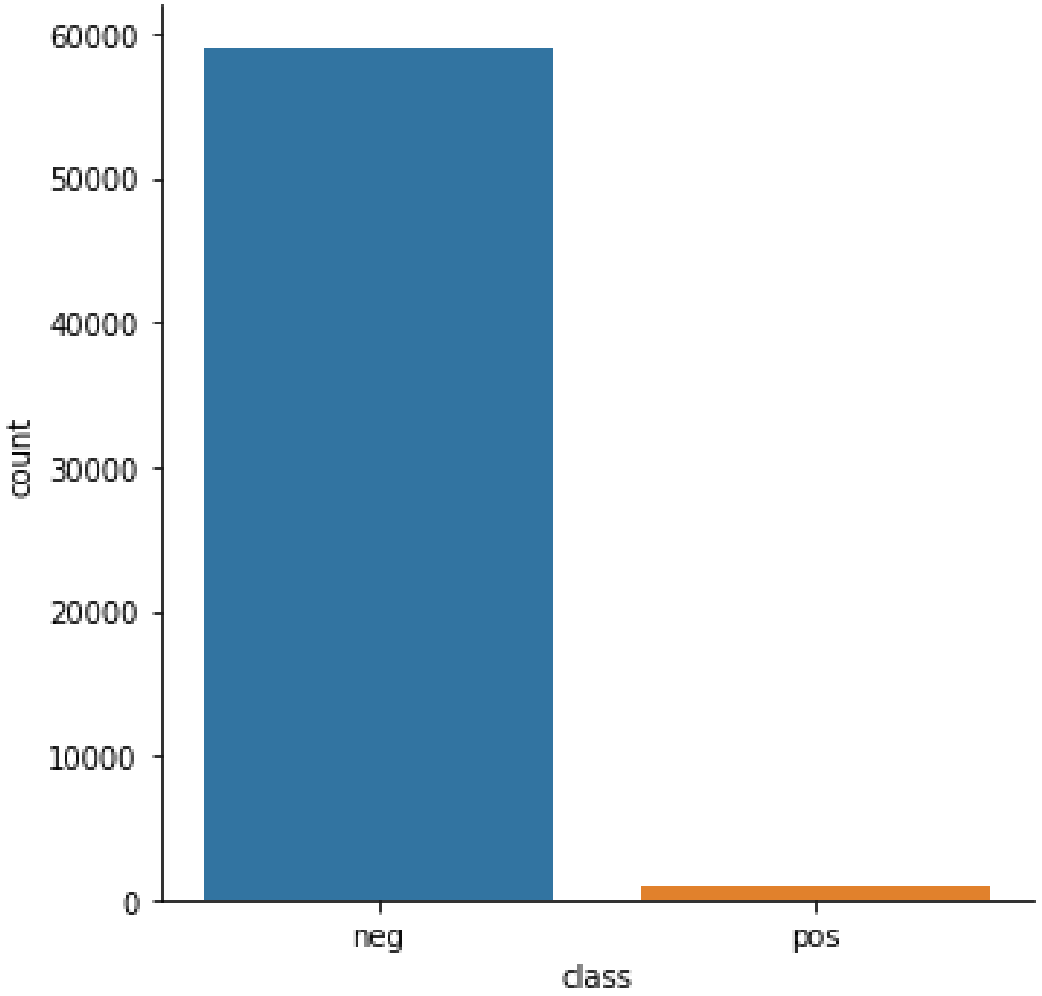
목차

Part 1

데이터

1. 데이터

EDA – 데이터 불균형

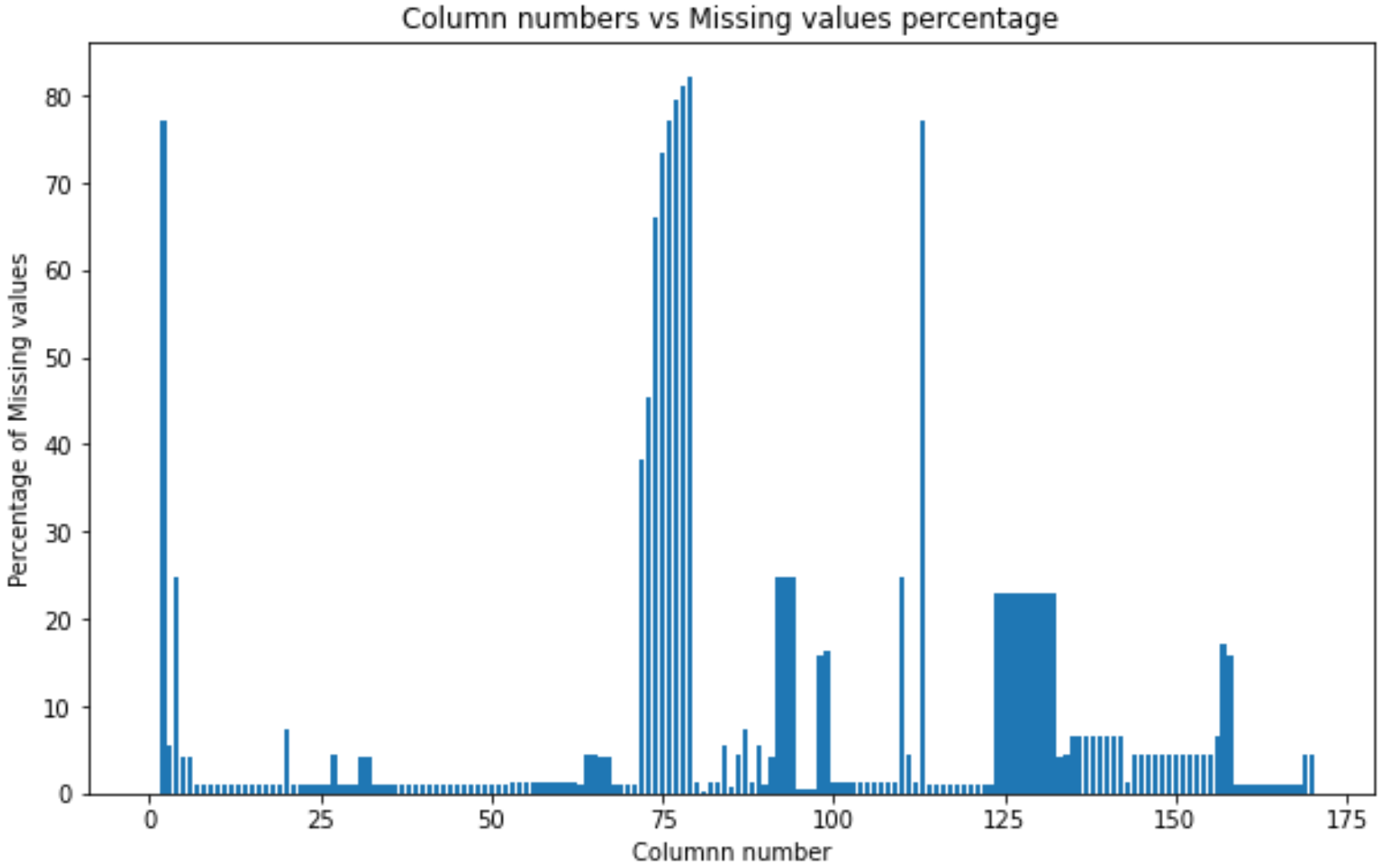


55934개의 negative

1066개의 positive

1. 데이터

EDA – 데이터 결측치



171개 feature
中
결측치 비율 20%이상
24개

Part 2

전처리

2. 전처리

결측치 처리

- Iterative Imputer
- KNN Imputer

Oversampling

- Adasyn
- SMOTE
- K-means SMOTE
- Borderline SMOTE

2. 전처리

결측치 처리 - Iterative Imputer (multivariate Imputer)

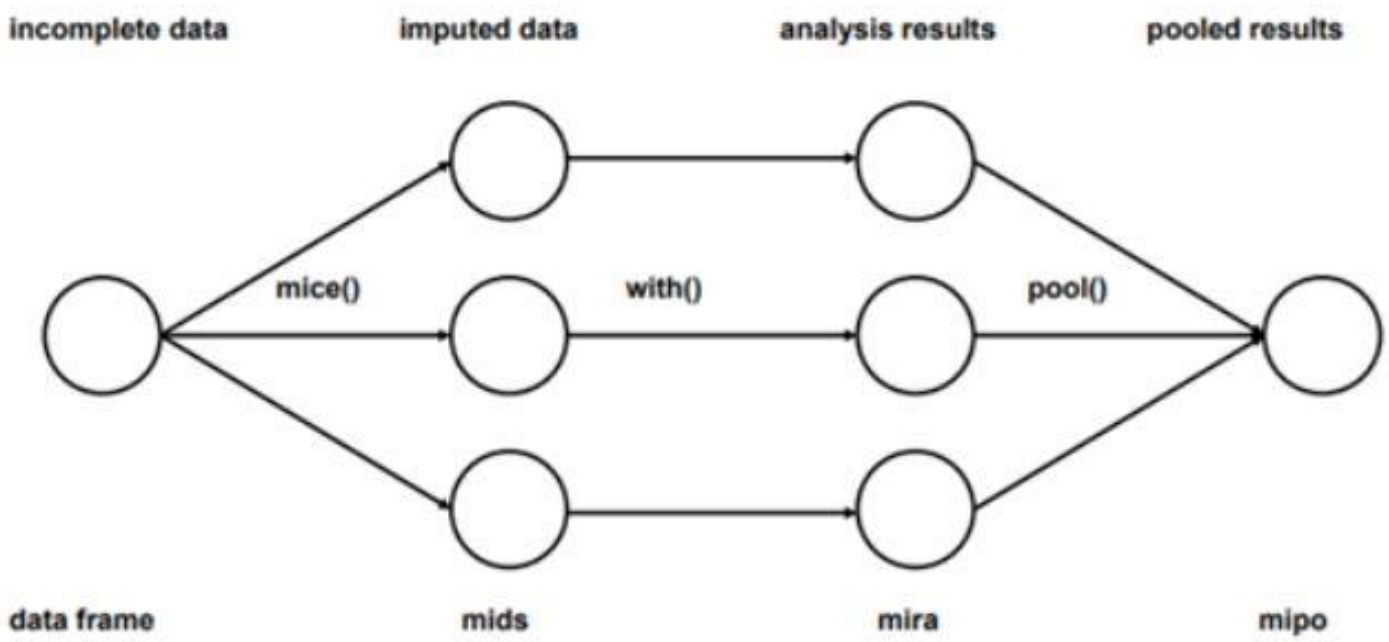
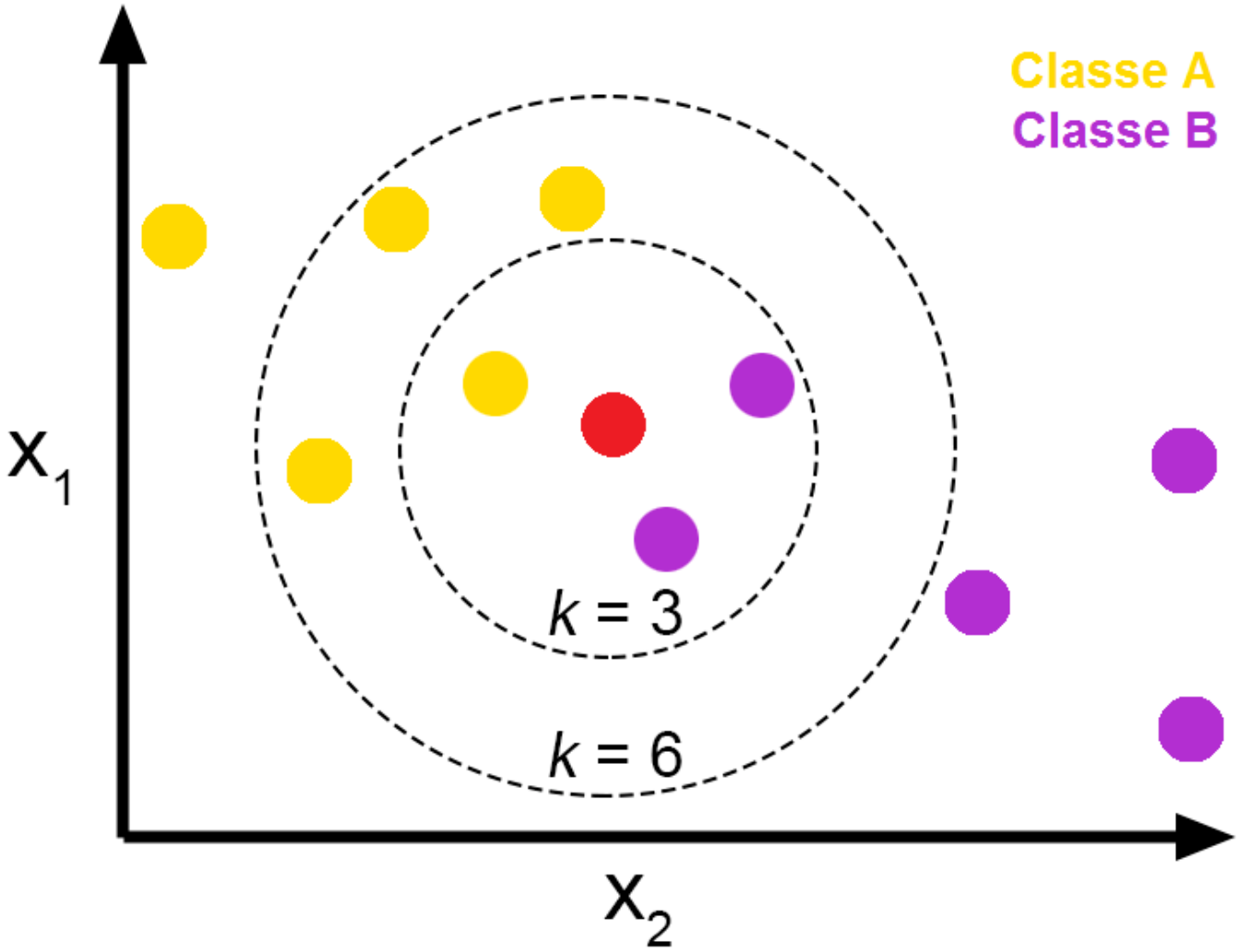


Figure 1: Main steps used in multiple imputation.

2. 전처리

결측치 처리 - KNN Imputer



- hyperparameter

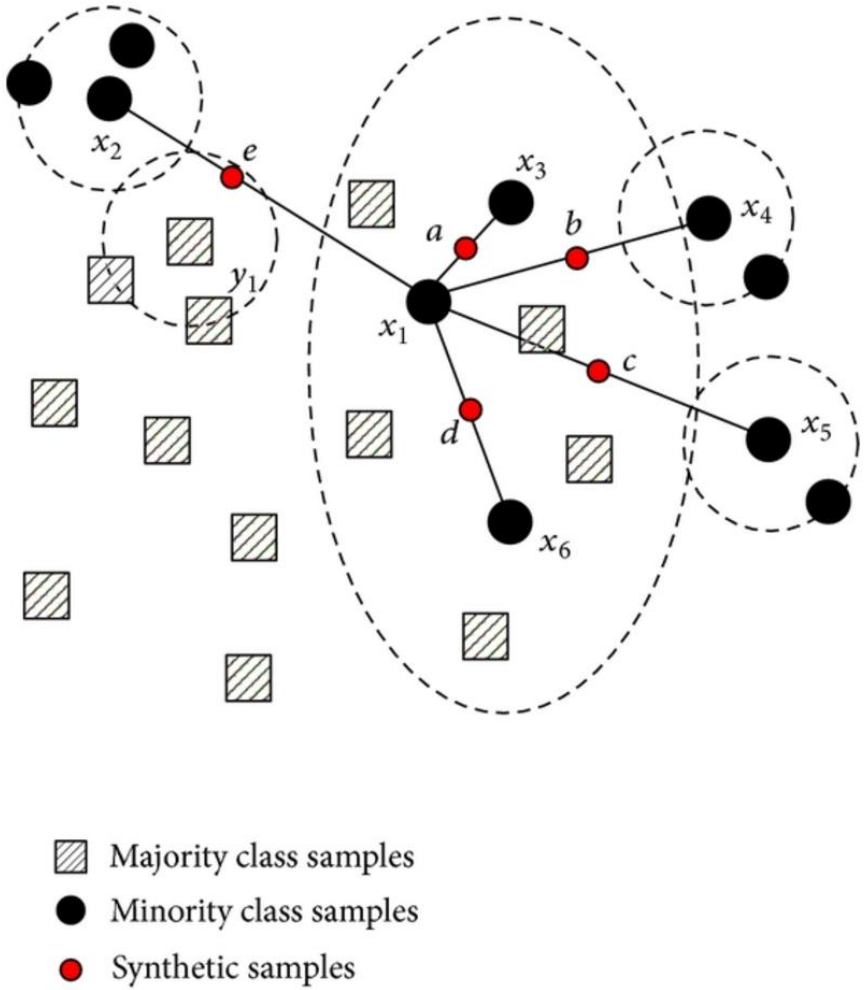
```
imputer = KNNImputer(n_neighbors=5,
weights="distance")
```

N = 5, 6, 7

Weights
=
uniform, distance

2. 전처리

Oversampling - SMOTE (k-nn기반)

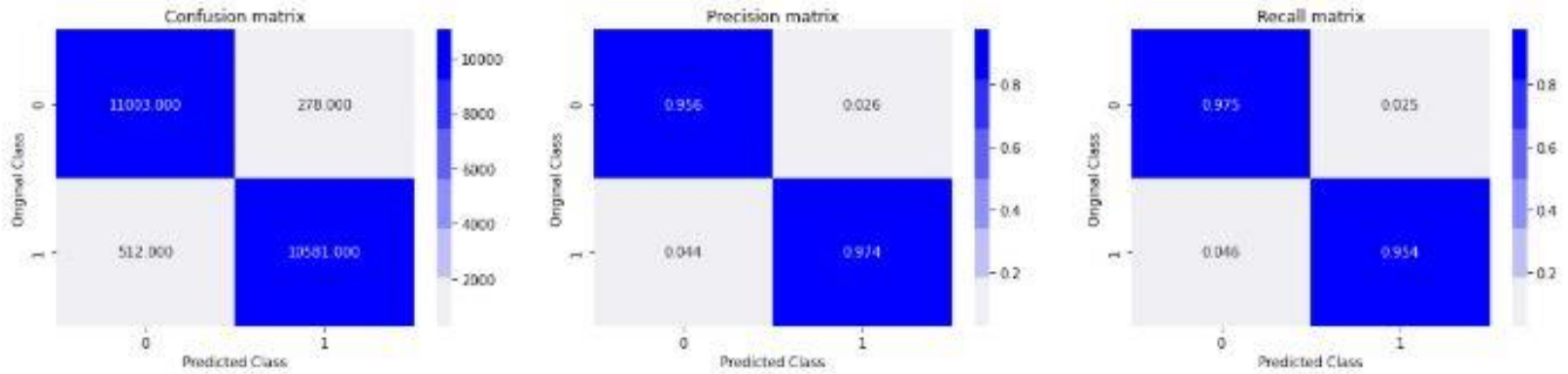


- Minority class에서 가장 가까운 이웃들을 찾음 - 군집 형성
- 그 후, 이웃들 사이에 선을 그어 그 선들에 무작위의 점을 생성

2. 전처리

Logistic Regression - SMOTE (k-nn기반)

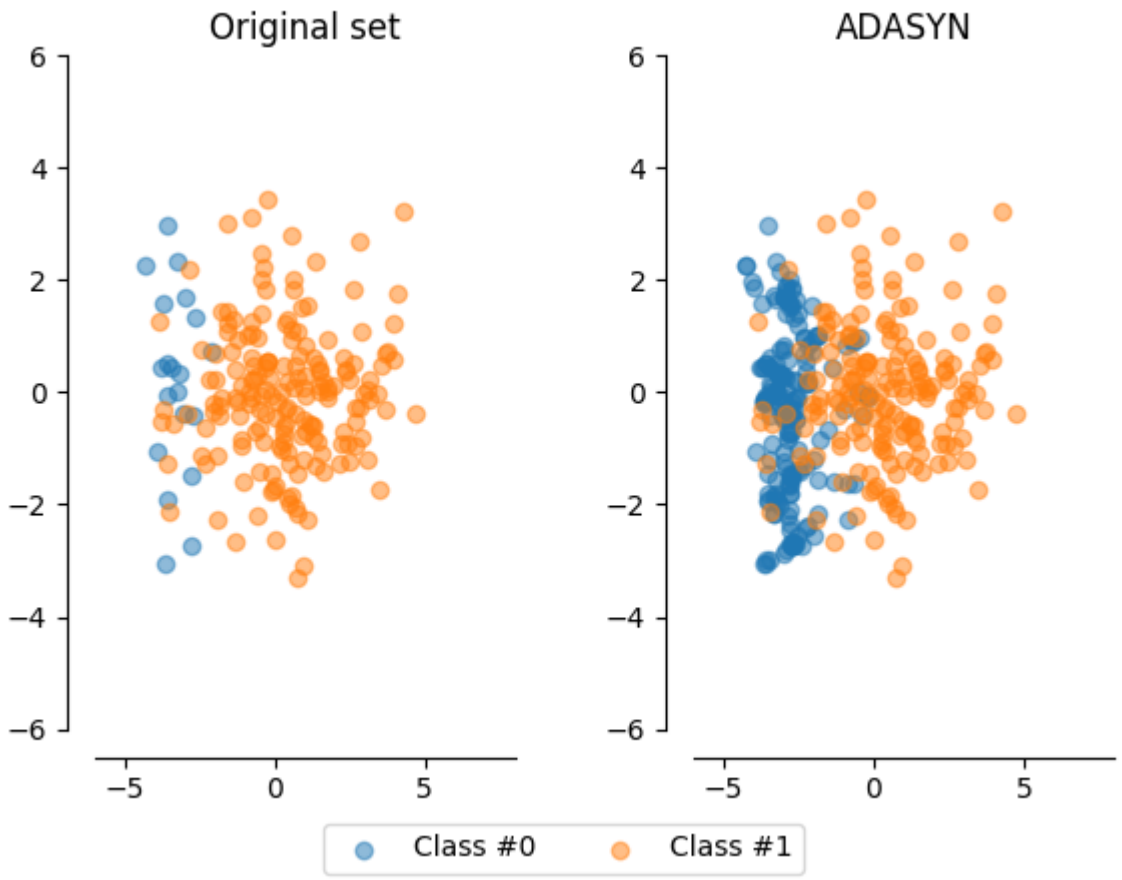
```
**************************************************
Test Accuracy is 0.9646911593814249
**************************************************
Test F1_score is 0.9640123906705541
**************************************************
Test AUC_score is 0.9646007807902841
**************************************************
```




```
None
Test cost is :  258780
**************************************************
```

2. 전처리

Oversampling - Adasyn

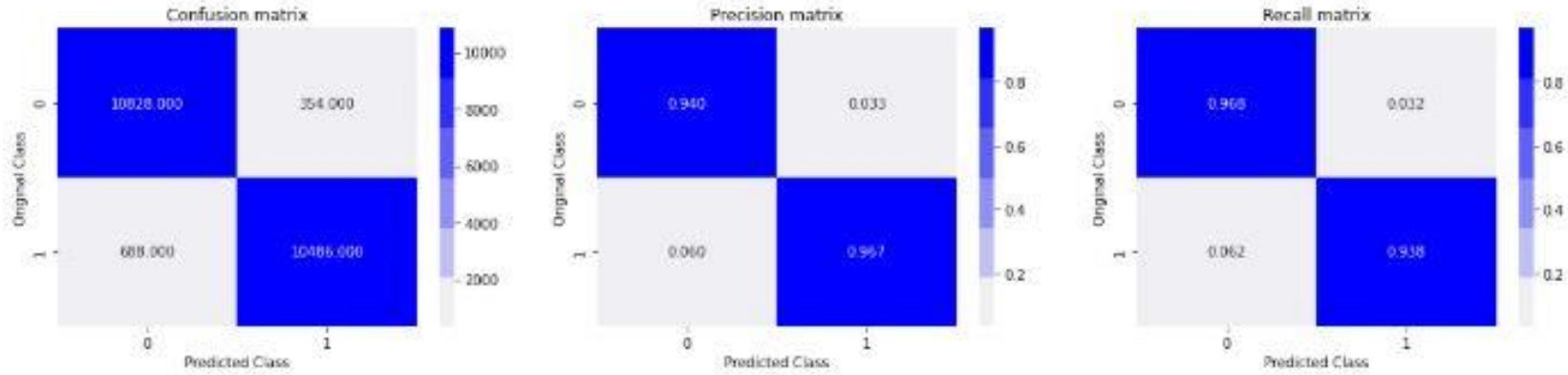


- SMOTE + 랜덤생성
- SMOTE를 할 때보다 큰 분산을 가져서 좀 더 현실적

2. 전처리

Logistic Regression - Adasyn

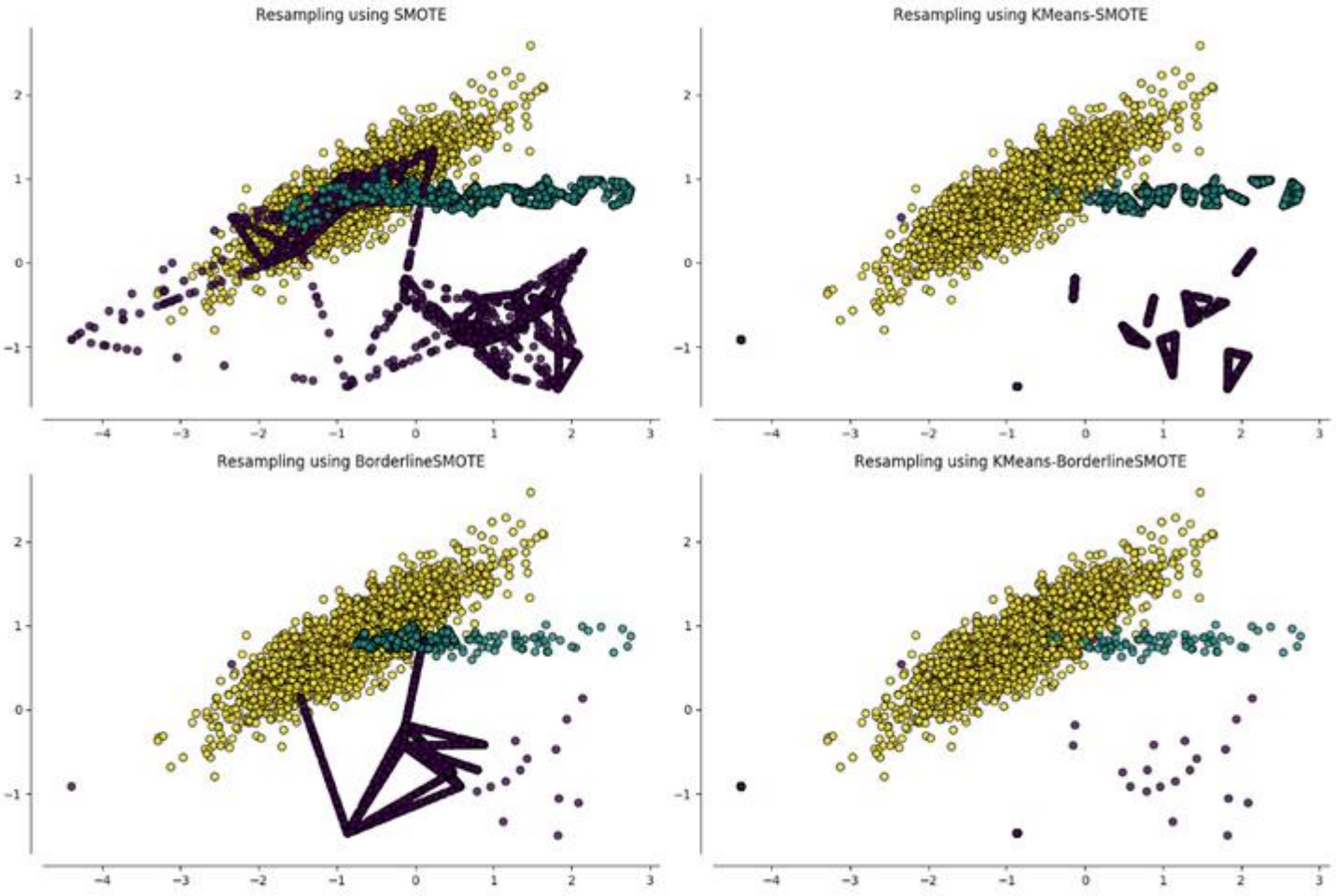
```
**************************************************
Test Accuracy is 0.9533905886562891
**************************************************
Test F1_score is 0.9526664849641137
**************************************************
Test AUC_score is 0.953385236449551 1
**************************************************
```




```
None
Test cost is :  347540
**************************************************
```

2. 전처리

Oversampling - K-means SMOTE

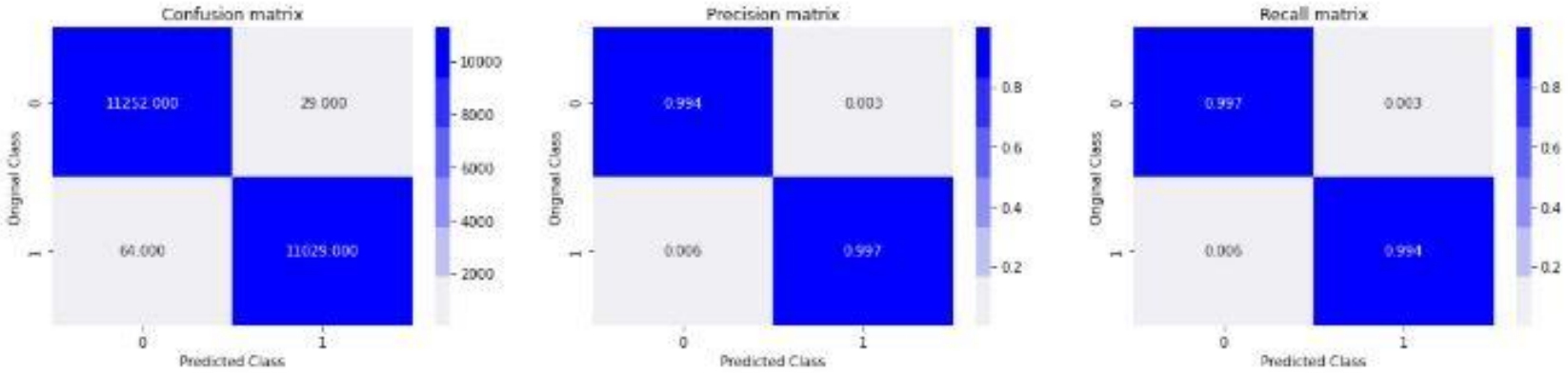


- 가까운 군집 별로 데이터를 생성하여, 기존 SMOTE가 noise를 많이 생성하는 단점을 보완

2. 전처리

Logistic Regression - K-means SMOTE

```
**************************************************
Test Accuracy is 0.9958433896486993
**************************************************
Test F1_score is 0.9958015439483545
**************************************************
Test AUC_score is 0.9958299508919333
**************************************************
```




```
None
Test cost is :  32290
**************************************************
```

2. 전처리

Oversampling - Borderline SMOTE

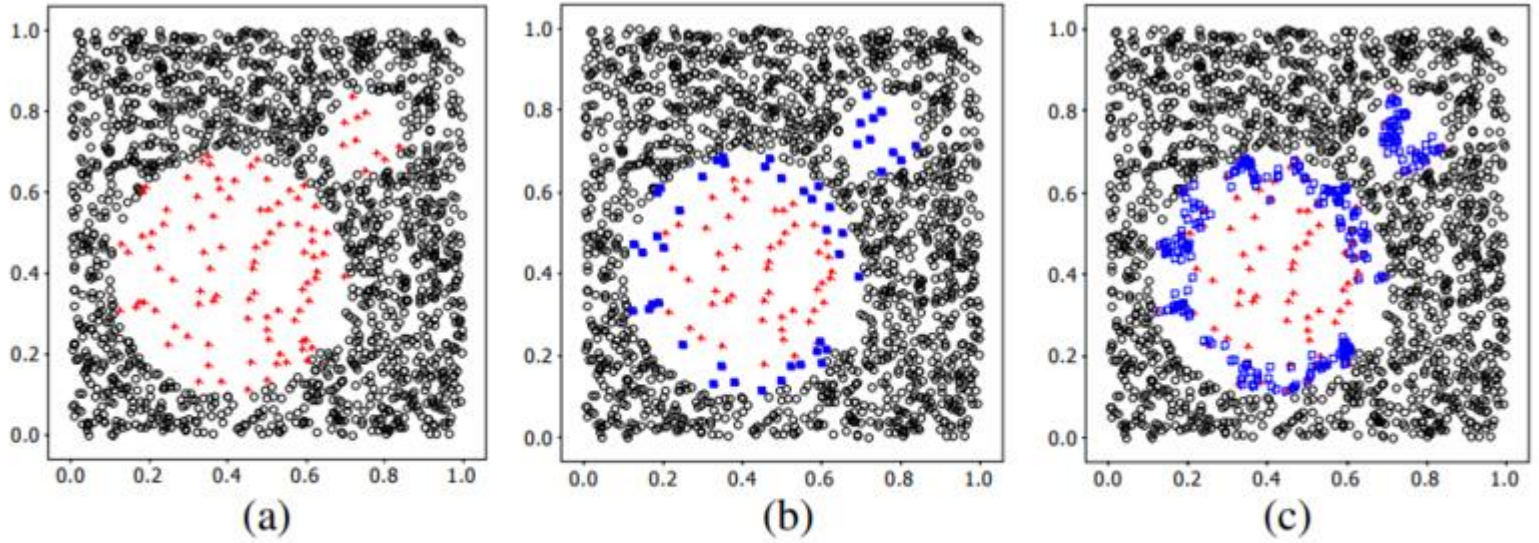


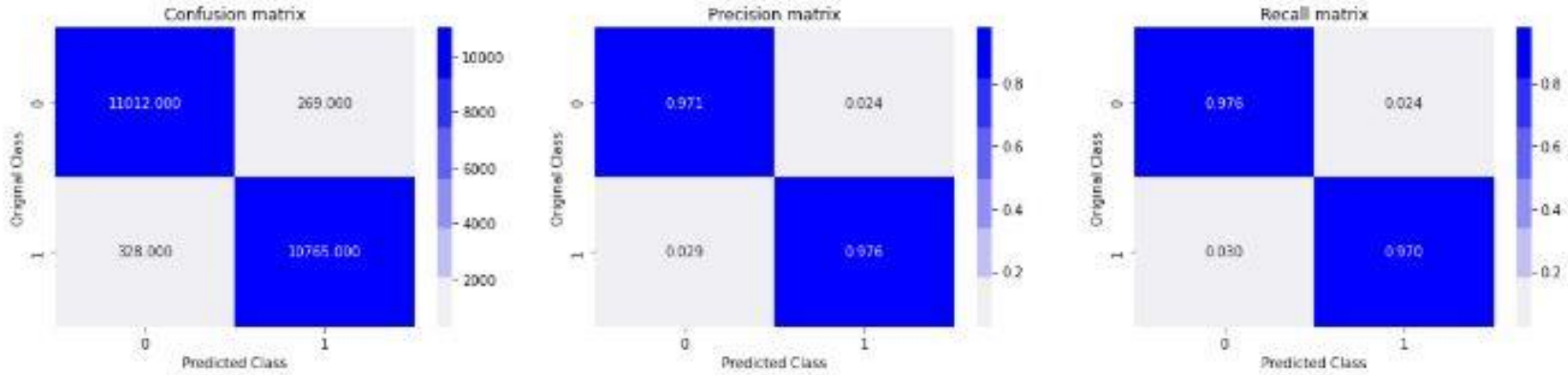
Fig. 1. (a) The original distribution of Circle data set. (b) The borderline minority examples (*solid squares*). (c) The borderline synthetic minority examples (*hollow squares*).

- 다른 class와의 경계(borderline)에 있는 샘플들을 늘림으로써 분류하기 어려운 부분에 집중

2. 전처리

Logistic Regression - Borderline SMOTE

```
**************************************************
Test Accuracy is 0.9733172432287477
**************************************************
Test F1_score is 0.9730193880779139
**************************************************
Test AUC_score is 0.9732932000319994
**************************************************
```




```
None
Test cost is :  166690
**************************************************
```

2. 전처리

나머지

- 데이터 분리

  데이터 셋을 train 데이터와 validation 데이터로 나눠줍니다.

  총 57000개의 데이터를 45600개의 train 데이터와 11400개의 validation 데이터로 나눠줍니다.

- Feature 제거

  총 171개의 feature 중에서 결측치 비율이 20%가 넘는 24개의 feature를 제거합니다.

  총 147개의 feture가 남습니다.

2. 전처리

8개의 전처리 데이터 선정

데이터 이름	Imputer	Oversampling	
NA1	Iterative imputer	Adasyn	
NA2	Iterative imputer	borderline smote	
NA3	Iterative imputer	borderline smote	+ SelectKbest (k = 144)
MJ1	knn imputer(uniform,k=5)	x	
MJ2	knn imputer(uniform,k=5)	K-means smote	
K5dist	knn imputer(distance,k=5)	x	
K6dist	knn imputer(distance,k=6)	x	
K7dist	knn imputer(distance,k=7)	x	

Part 3

모델링

3. 모델링

Model

- MLP
- CNN
- FCN

Activation

- Sigmoid
- Softmax
- ReLU

Loss

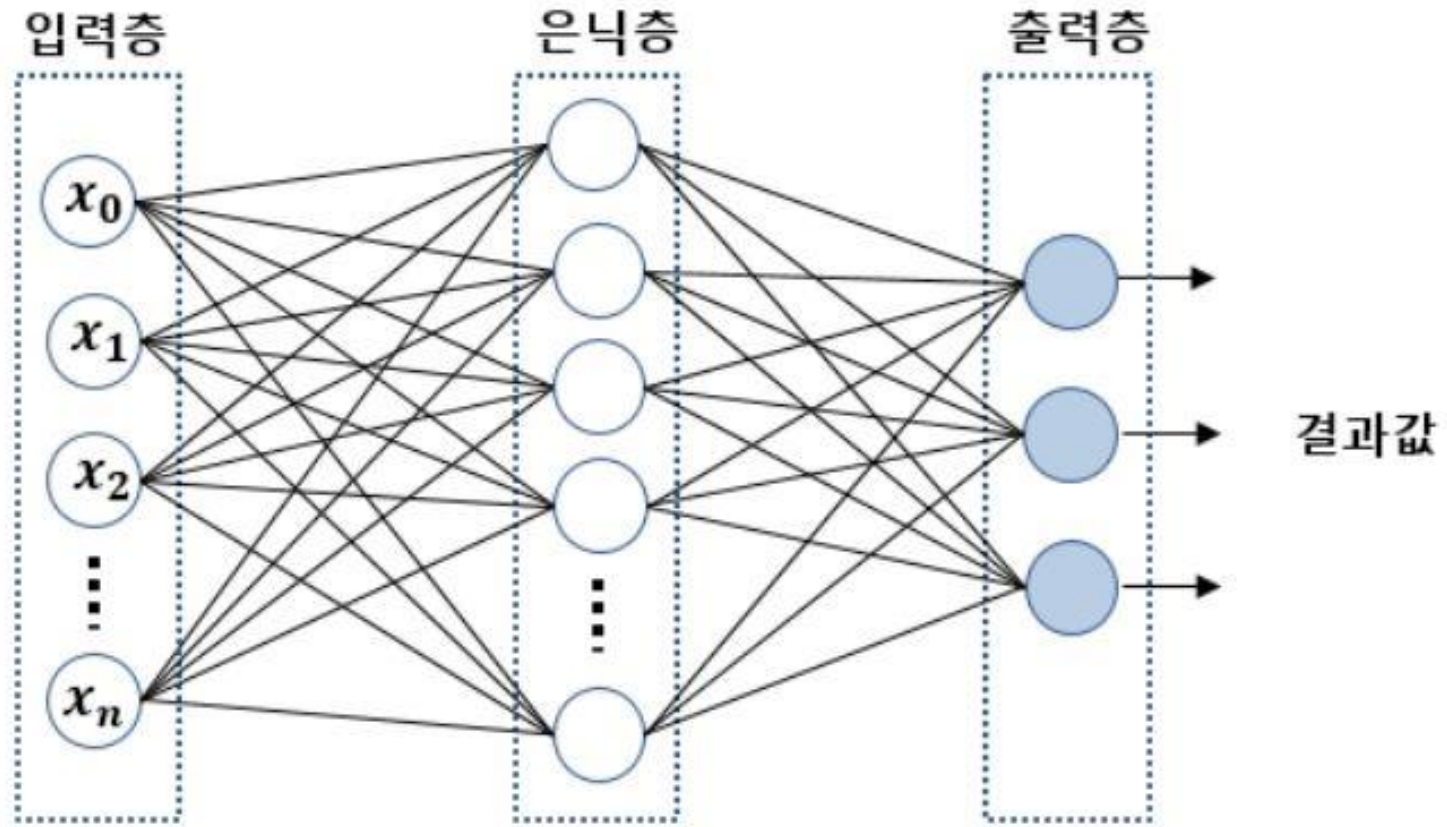
- CrossEntropy
- BinaryCrossEntropy

Optimizer

- SGD + Momentum
- Adam
- RMSprop

3. 모델링

MLP (Multi-Layer Perceptron)



3. 모델링

MLP - hyperparameter tuning

- Activation
- Batch_size
- Layer 수
- Dropout1
- Dropout2
- Epoch
- Initializer
- Optimizer
- Units1
- Units2

- Activation : relu
- Batch_size : 128
- Layer 수 : 2
- Dropout1 : 0.0
- Dropout2 : 0.2
- Epoch : 10
- Initializer : lecun_uniform
- Optimizer : adam
- Units1 : 256
- Units2 : 32

3. 모델링

MLP - 결과

Oversampling O

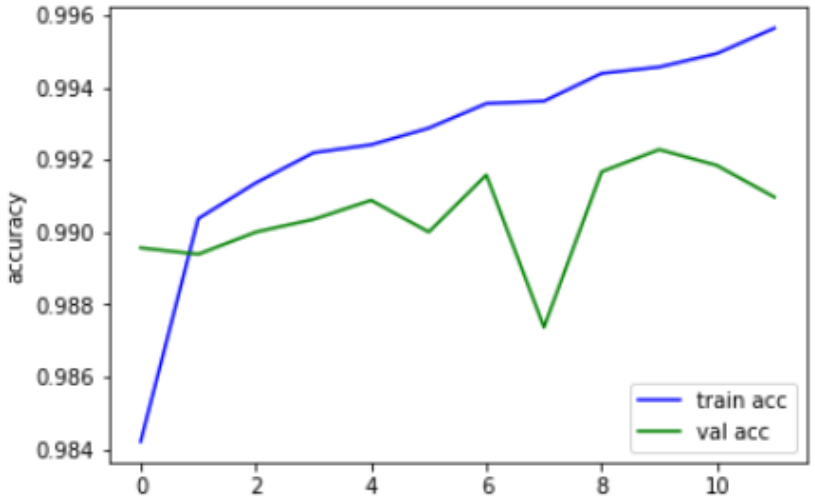
NA1	NA2	NA3	MJ2
0.66	0.78	0.16	1.9

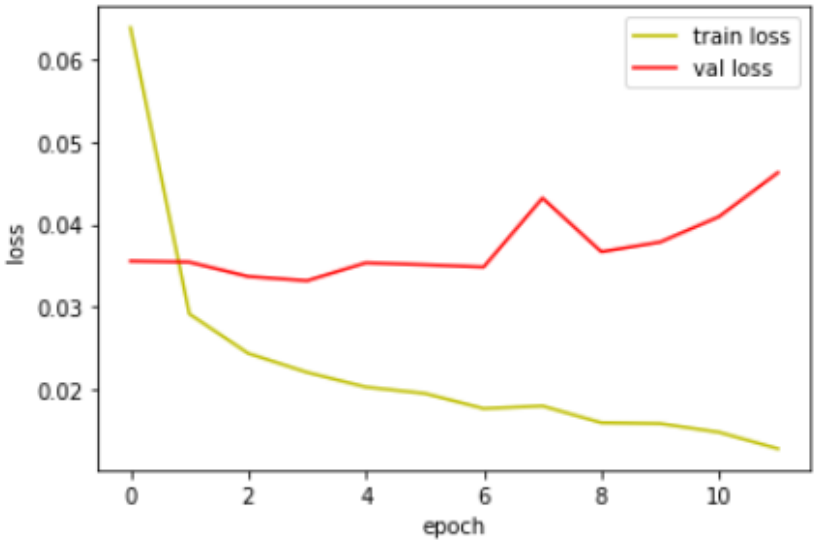
Oversampling X

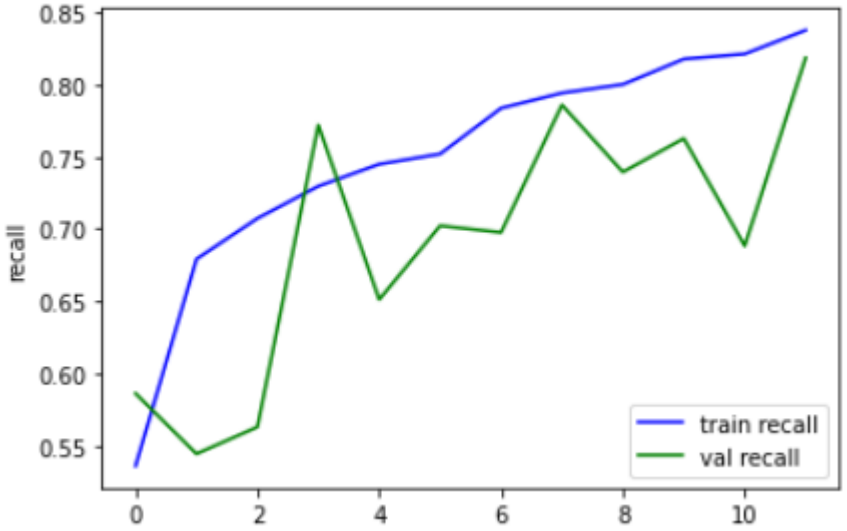
MJ1	K5dist	K6dist	K7dist
1.85	1.77	3.28	3.31

3. 모델링

MLP - 결과(K5dist)





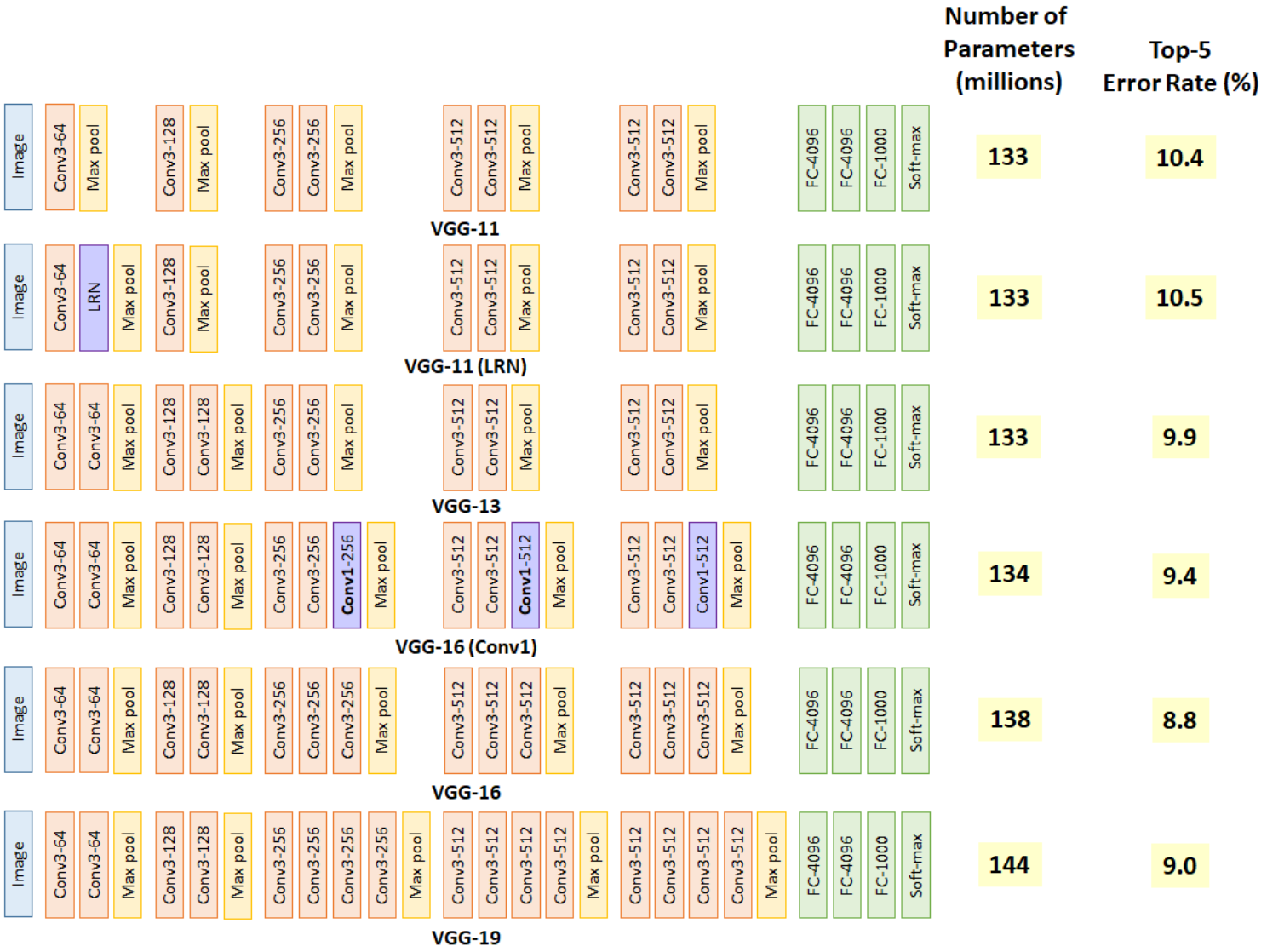


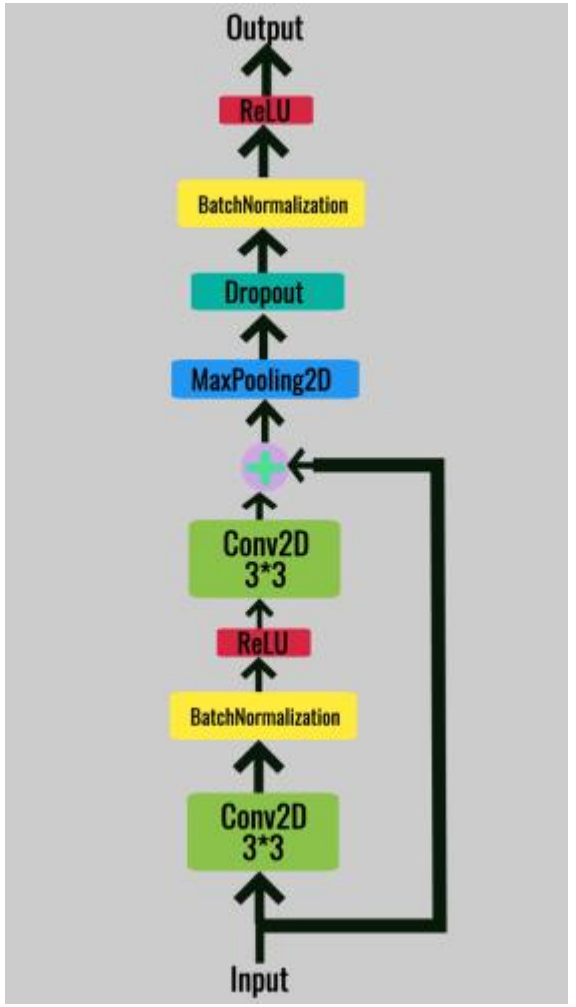
- Total cost : 20140

- Average cost : 1.77

3. 모델링

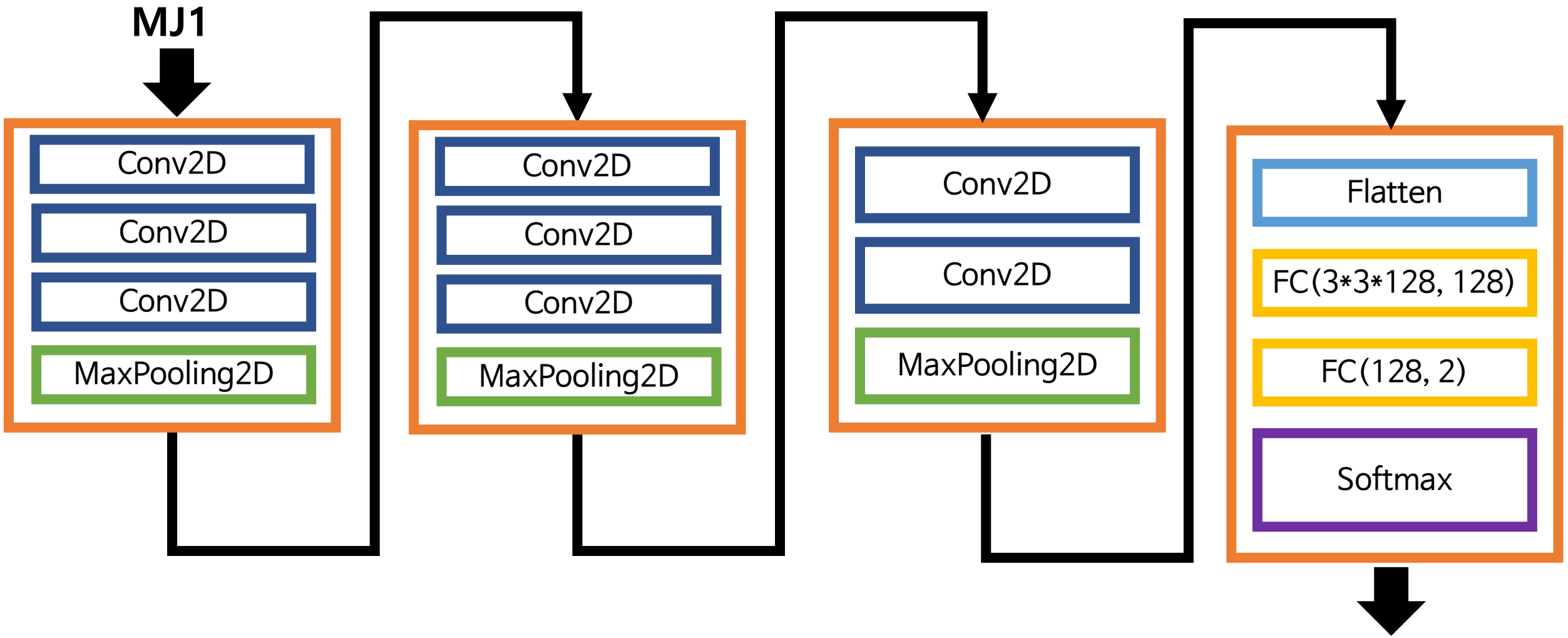
CNN (Convolutional Neural Network)





3. 모델링

CNN (Convolutional Neural Network)



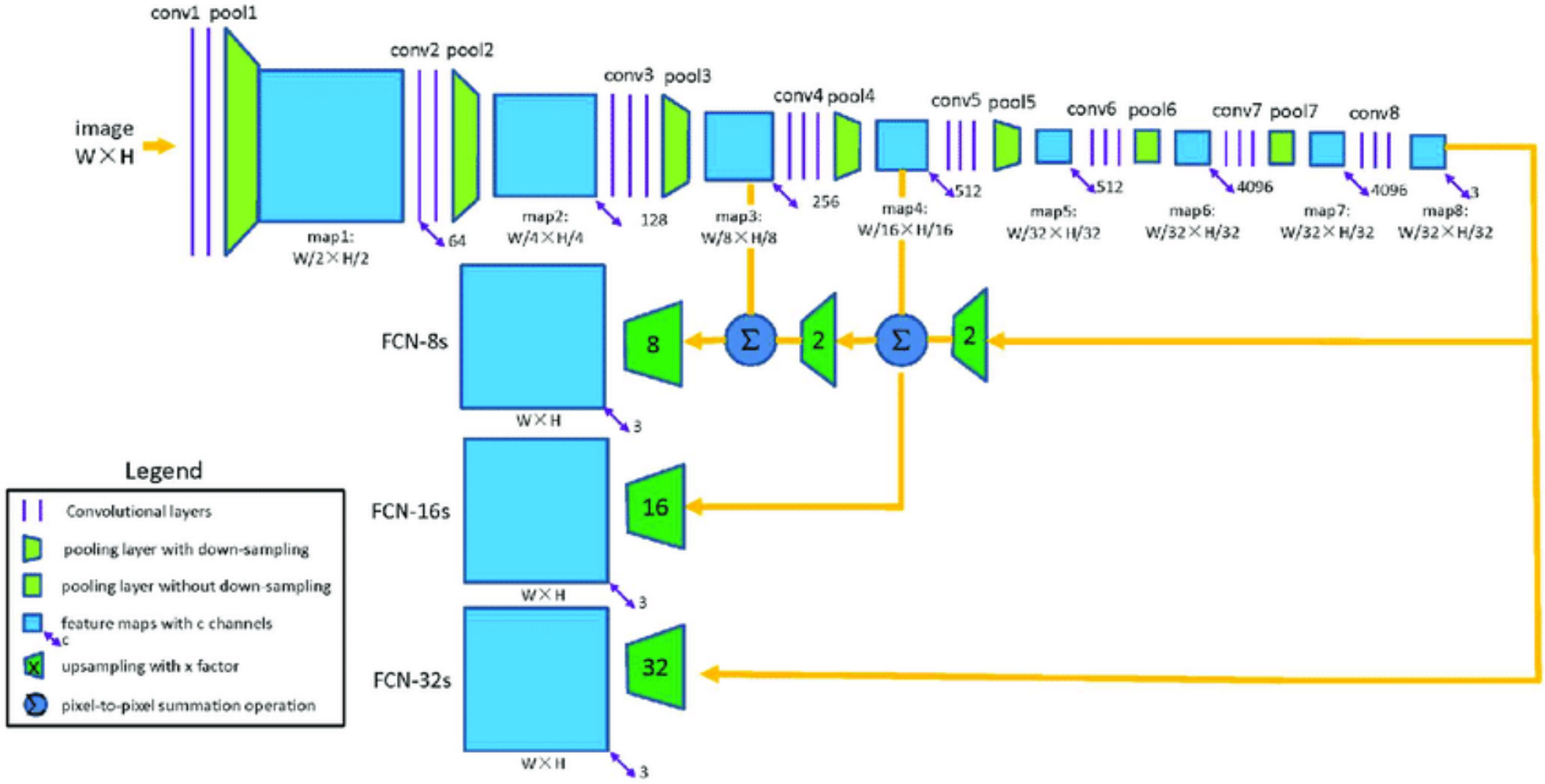
3. 모델링

CNN 모델 학습 결과

모 델 명	Train Total Cost	Validation Total Cost
CNN (loss:CrossEntropy activation:Softmax optimizer: SGD+Momentum)	61.6115	264.1665
CNN (loss:BinaryCrossEntropy activation:Sigmoid optimizer:SGD+Momentum)	9.2105	38.6400
CNN (loss:CrossEntropy activation:Softmax optimizer: Adam)	9.1885	36.4035
CNN (loss:BinaryCrossEntropy activation:Sigmoid optimizer:Adam)	9.1885	39.3855

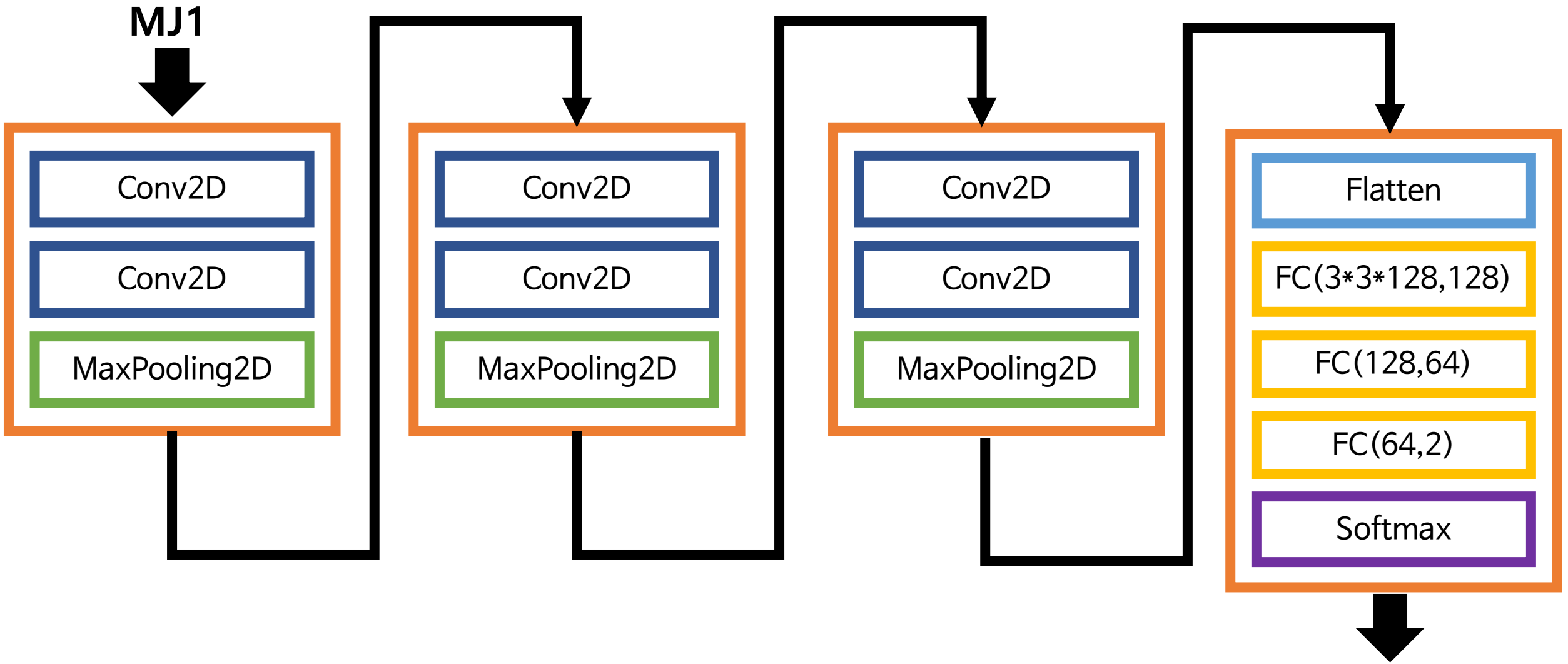
3. 모델링

FCN


conv1 pool1
conv2 pool2
conv3 pool3
conv4 pool4
conv5 pool5
conv6 pool6
conv7 pool7
conv8
image
W×H
map1:
W/2×H/2
64
map2:
W/4×H/4
128
map3:
W/8×H/8
256
map4:
W/16×H/16
512
map5:
W/32×H/32
512
map6:
W/32×H/32
4096
map7:
W/32×H/32
4096
map8:
W/32×H/32
3
FCN-8s
FCN-16s
FCN-32s
W×H
3
8
16
32
2
2
Σ
Σ
Legend
Convolutional layers
pooling layer with down-sampling
pooling layer without down-sampling
feature maps with c channels
c
upsampling with x factor
pixel-to-pixel summation operation

3. 모델링

FCN


MJ1
Conv2D
Conv2D
MaxPooling2D
Conv2D
Conv2D
MaxPooling2D
Conv2D
Conv2D
MaxPooling2D
Flatten
FC(3*3*128,128)
FC(128,64)
FC(64,2)
Softmax

3. 모델링

FCN 모델 학습 결과

모 델 명	Train Total Cost	Validation Total Cost
FCN (loss:CrossEntropy activation:Softmax optimizer: SGD+Momentum)	9.1995	38.7715
FCN (loss:BinaryCrossEntropy activation:Sigmoid optimizer:SGD+Momentum)	9.2105	38.9035
FCN (loss:CrossEntropy activation:Softmax optimizer: Adam)	9.1995	37.5875
FCN (loss:BinaryCrossEntropy activation:Sigmoid optimizer:Adam)	9.1995	38.8595

Part 4

결론

4. 결론

해석

하이퍼파라미터 튜닝 결론

- Activation function은 relu / Optimizer는 adam이 가장 성능이 좋음
- Dropout은 대체로 없는 것이 좋음
- 층 개수와 성능이 비례하지 않음. 성능이 좋은 모델들은 층을 2개만 쌓아도 잘 나옴.
- Batch size는 작을 때보다 클 때 (128) 성능이 좋음
- 가중치 초기화는 대체로 uniform이 normal보다 성능이 좋음

4. 결론

해석

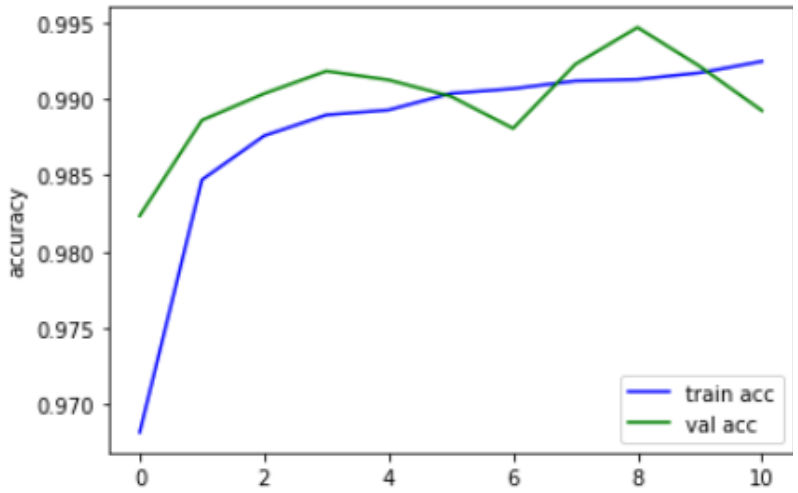
Imputation < Imputation + **Oversampling**

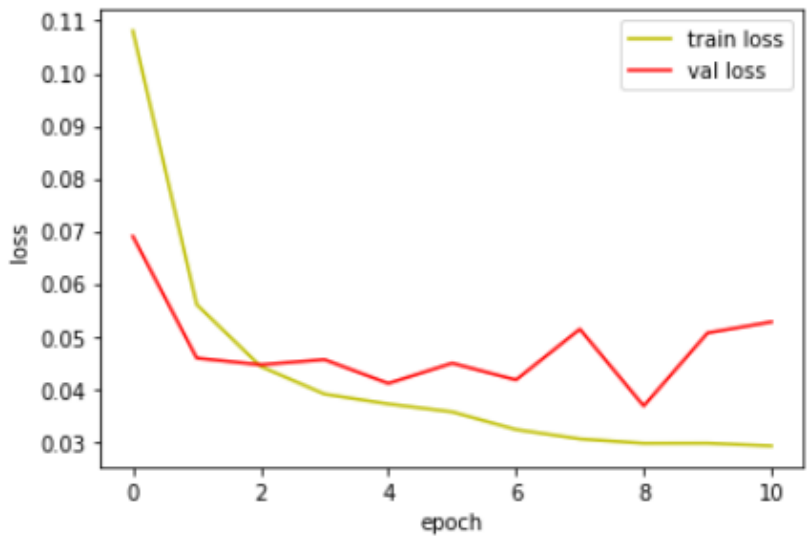
- **MLP에서는 오버 샘플링까지 한 데이터가 성능이 좋았음.**
- 오버샘플링을 하지 않은 경우에는, 학습할 때 성능이 들쭉날쭉하고 데이터마다 결과가 상이
- 성능, 안정성 모든 면에서 좋은 오버샘플링 방식을 선택하는 것이 적합

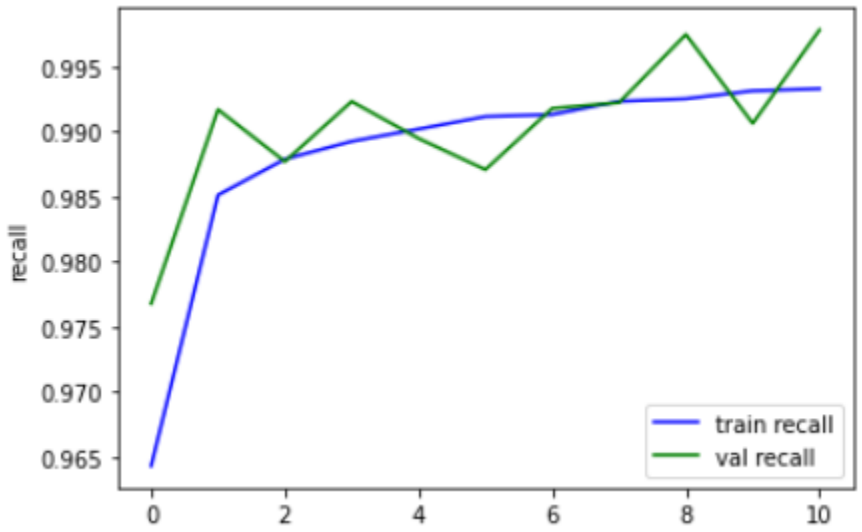
4. 결론

해석

NA1 - (Oversampling O) 학습 그래프



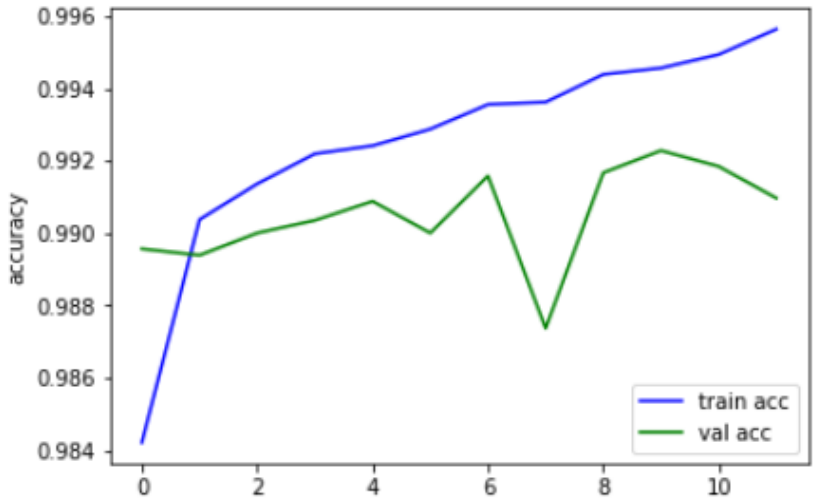


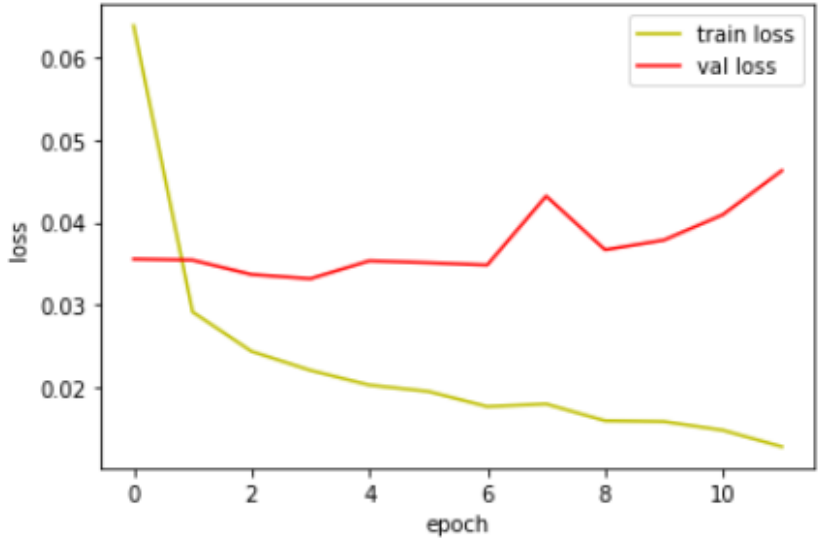


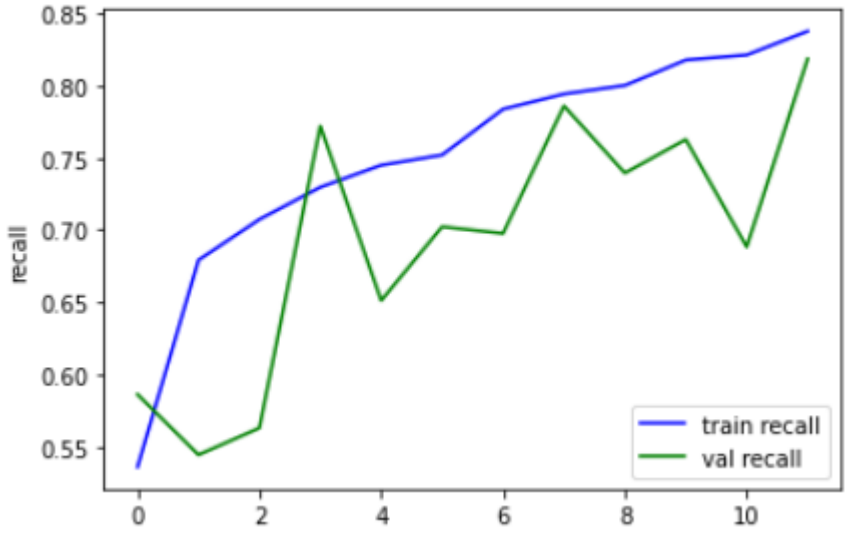
4. 결론

해석

K5dist - (Oversampling X) 학습 그래프







4. 결론

해석

- 실제로, 오버샘플링 데이터에서 성능이 좋았던 모델(NA1 – RE)과 오버샘플링 없이 Imputation만 진행한 데이터에서 성능이 좋았던 모델 (k5dist)의 학습 그래프를 비교해보면 **오버샘플링 데이터가 안정적**

4. 결론

해석

오버샘플링 데이터셋의 성능이 좋은 이유는?

- **비대칭 데이터** 셋에서는 **정확도가 높아도, recall이 급격히 작아지는 현상** 발생
- 하지만, 우리 데이터는 FN에 대한 가중치가 크고, 따라서 recall (실제 positive class를 맞게 예측한 비율)이 중요

  - neg를 pos로 분류하는건 10달러의 비용
  - pos를 neg로 분류하는건 500 달러의 비용

Cost = (10 * FP) + (500 * FN)

$$(Recall) = \frac{TP}{TP + FN}$$

- 또한, **신경망 모델**의 특성상 데이터가 많이 확보될수록 유리

4. 결론

Test data

- Confusion Matrix

18586	105
69	240

- Accuracy : 0.9908421052631579
- Confusion Matrix : 35550

감 사 합 니 다