▼ **그룹별 통계치를 활용한 고급 피처 엔지니어링 버전**

1. 전처리 (Preprocessing)

- **데이터 분리:** 학습 데이터셋(`train.csv`)을 훈련(80%) 및 검증(20%) 데이터로 분리했습니다. 이때 `support_needs`라는 타겟 변수의 클래스 비율을 유지하기 위해 `stratify` 옵션을 사용했습니다.
- **심화 피처 엔지니어링 (Advanced Feature Engineering):**
  - `gender`, `subscription_type`, `contract_length`를 기준으로 그룹을 나누고, 각 그룹 내의 `age`, `tenure`, `frequent`, `after_interaction` 수치에 대한 평균(`_mean`)과 표준편차(`_std`)를 계산
  - 이 그룹 통계치를 활용해 두 가지 새로운 피처를 생성:
    - **그룹 평균과의 차이(`_diff_from_..._mean`):** 개인의 값이 속한 그룹의 평균과 얼마나 다른지를 나타냅니다. 예를 들어, 특정 성별 그룹의 평균 나이보다 해당 개인의 나이가 얼마나 많은지를 알 수 있습니다.
    - **그룹 표준편차로 정규화된 값(`_norm_by_...`):** 그룹 내에서 해당 개인의 수치가 얼마나 표준에서 벗어나 있는지를 나타내, 이상치 경향을 파악하는 데 유용합니다.
  - 이렇게 생성된 새로운 피처들은 모델이 각 고객의 특성을 해당 그룹과 비교하여 더 심층적으로 이해하도록 돕습니다.
- **데이터 스케일링 & 인코딩:**
  - **수치형 데이터:** `StandardScaler`를 사용해 피처 엔지니어링으로 추가된 새로운 변수들을 포함한 모든 수치형 변수를 표준화(평균 0, 표준편차 1)했습니다.
  - **범주형 데이터:** `gender`와 `subscription_type` 변수는 `pd.get_dummies`를 사용해 **원-핫 인코딩**으로 변환했습니다.

2. 모델 (Model)

- **모델 구조: WideMLP** (다층 퍼셉트론) 모델을 사용
  - 4개의 은닉층과 하나의 출력층으로 구성
  - 각 은닉층에는 **배치 정규화(`BatchNorm1d`)**, **ReLU 활성화 함수**, 그리고 드롭아웃(`Dropout`)이 순차적으로 적용
- **파라미터:**

- 은닉층의 노드(뉴런) 수는 **512개**로 설정
- 드롭아웃 비율은 **0.2**

3. 하이퍼파라미터 및 학습 설정

- **손실 함수:** `FocalLoss`를 사용하여, 클래스 불균형 문제를 해결
- **옵티마이저: AdamW** 옵티마이저를 사용, 초기 학습률은 3×10^{-4}로 설정
- **학습률 스케줄러:** `CosineAnnealingWarmRestarts`를 적용하여 학습률을 주기적으로 조절함으로써, 학습이 지역 최적해에 빠지는 것을 방지
- **조기 종료 (`Early Stopping`):** 검증 손실(`ValLoss`)이 **15 에포크** 동안 개선되지 않으면 학습을 중단하고, 최적의 모델 가중치를 저장하도록 설정 실제로 로그를 보면 **50번째 에포크**에서 조기 종료가 발생

```
🔥 모델 학습을 시작합니다...
Epoch 001 | TrainLoss 0.5072 | TrainAcc 0.4181 | ValLoss 0.4656 | ValAcc 0.4576 | MacroF1 0.4404 | LR 2.93e-04
Epoch 002 | TrainLoss 0.4785 | TrainAcc 0.4385 | ValLoss 0.4668 | ValAcc 0.4606 | MacroF1 0.4404 | LR 2.71e-04
Epoch 003 | TrainLoss 0.4718 | TrainAcc 0.4424 | ValLoss 0.4631 | ValAcc 0.4718 | MacroF1 0.4457 | LR 2.38e-04
Epoch 004 | TrainLoss 0.4662 | TrainAcc 0.4567 | ValLoss 0.4636 | ValAcc 0.4614 | MacroF1 0.4503 | LR 1.97e-04
Epoch 005 | TrainLoss 0.4622 | TrainAcc 0.4606 | ValLoss 0.4579 | ValAcc 0.4689 | MacroF1 0.4591 | LR 1.50e-04
Epoch 006 | TrainLoss 0.4574 | TrainAcc 0.4665 | ValLoss 0.4581 | ValAcc 0.4770 | MacroF1 0.4605 | LR 1.04e-04
Epoch 007 | TrainLoss 0.4530 | TrainAcc 0.4688 | ValLoss 0.4553 | ValAcc 0.4749 | MacroF1 0.4612 | LR 6.26e-05
Epoch 008 | TrainLoss 0.4522 | TrainAcc 0.4725 | ValLoss 0.4554 | ValAcc 0.4783 | MacroF1 0.4613 | LR 2.96e-05
Epoch 009 | TrainLoss 0.4510 | TrainAcc 0.4738 | ValLoss 0.4555 | ValAcc 0.4765 | MacroF1 0.4612 | LR 8.32e-06
Epoch 010 | TrainLoss 0.4520 | TrainAcc 0.4746 | ValLoss 0.4559 | ValAcc 0.4765 | MacroF1 0.4618 | LR 3.00e-04
Epoch 011 | TrainLoss 0.4557 | TrainAcc 0.4636 | ValLoss 0.4566 | ValAcc 0.4989 | MacroF1 0.4538 | LR 2.98e-04
Epoch 012 | TrainLoss 0.4519 | TrainAcc 0.4717 | ValLoss 0.4526 | ValAcc 0.4911 | MacroF1 0.4607 | LR 2.93e-04
Epoch 013 | TrainLoss 0.4483 | TrainAcc 0.4747 | ValLoss 0.4523 | ValAcc 0.4827 | MacroF1 0.4642 | LR 2.84e-04
Epoch 014 | TrainLoss 0.4441 | TrainAcc 0.4856 | ValLoss 0.4548 | ValAcc 0.4697 | MacroF1 0.4603 | LR 2.71e-04
Epoch 015 | TrainLoss 0.4431 | TrainAcc 0.4883 | ValLoss 0.4506 | ValAcc 0.4851 | MacroF1 0.4549 | LR 2.56e-04
Epoch 016 | TrainLoss 0.4398 | TrainAcc 0.4898 | ValLoss 0.4529 | ValAcc 0.4747 | MacroF1 0.4619 | LR 2.38e-04
Epoch 017 | TrainLoss 0.4371 | TrainAcc 0.4976 | ValLoss 0.4534 | ValAcc 0.4749 | MacroF1 0.4622 | LR 2.18e-04
Epoch 018 | TrainLoss 0.4368 | TrainAcc 0.4964 | ValLoss 0.4496 | ValAcc 0.4789 | MacroF1 0.4647 | LR 1.97e-04
Epoch 019 | TrainLoss 0.4367 | TrainAcc 0.4966 | ValLoss 0.4495 | ValAcc 0.4738 | MacroF1 0.4624 | LR 1.74e-04
Epoch 020 | TrainLoss 0.4351 | TrainAcc 0.4991 | ValLoss 0.4492 | ValAcc 0.4710 | MacroF1 0.4621 | LR 1.50e-04
Epoch 021 | TrainLoss 0.4340 | TrainAcc 0.5015 | ValLoss 0.4492 | ValAcc 0.4809 | MacroF1 0.4695 | LR 1.27e-04
Epoch 022 | TrainLoss 0.4320 | TrainAcc 0.5027 | ValLoss 0.4495 | ValAcc 0.4856 | MacroF1 0.4667 | LR 1.04e-04
Epoch 023 | TrainLoss 0.4321 | TrainAcc 0.5063 | ValLoss 0.4499 | ValAcc 0.4759 | MacroF1 0.4643 | LR 8.26e-05
Epoch 024 | TrainLoss 0.4308 | TrainAcc 0.5038 | ValLoss 0.4497 | ValAcc 0.4841 | MacroF1 0.4694 | LR 6.26e-05
...
Epoch 048 | TrainLoss 0.4253 | TrainAcc 0.5081 | ValLoss 0.4508 | ValAcc 0.4843 | MacroF1 0.4677 | LR 1.74e-04
Epoch 049 | TrainLoss 0.4243 | TrainAcc 0.5127 | ValLoss 0.4473 | ValAcc 0.4705 | MacroF1 0.4613 | LR 1.62e-04
Epoch 050 | TrainLoss 0.4245 | TrainAcc 0.5114 | ValLoss 0.4484 | ValAcc 0.4875 | MacroF1 0.4676 | LR 1.50e-04
> 50 Epoch에서 조기 종료. Best ValLoss: 0.4458
```

MacroF1 0.4676

▼ **Wide & Deep 모델 사용**

전처리 (Preprocessing)

Wide & Deep 모델에 특화된 2단계 전처리 과정.

- **피처 엔지니어링:**

    - **비율 피처 생성:** `tenure`(총 이용 기간), `frequent`(서비스 이용일), `after_interaction`(최근 이용 경과 기간) 변수를 조합하여 비율 형태의 새로운 피처들을 생성
        - `tenure_freq_ratio`: 총 이용 기간 대비 서비스 이용일 비율.
        - `tenure_inter_ratio`: 총 이용 기간 대비 최근 이용 경과 기간 비율.
        - `freq_inter_ratio`: 서비스 이용일 대비 최근 이용 경과 기간 비율.
- **Wide & Deep 전처리:**
    - **Deep 파트:** `age`, `tenure`, `frequent` 등 **연속적인 수치형 피처**는 `StandardScaler`로 표준화합니다. `gender`, `subscription_type`, `contract_length` 등 **범주형 피처**는 `OneHotEncoder`로 변환합니다.
    - **Wide 파트:** `subscription_type`과 `contract_length`를 조합하여 `sub_x_contract`와 같은 교차 특성(Cross-Features)을 생성한 후, `OneHotEncoder`를 통해 변환
- **클래스 가중치**

모델 (Model)

Wide & Deep 모델

- **구조:**
    - **Deep 파트:** 여러 Dense 레이어와 Dropout 레이어로 구성되어 **복잡한 패턴**을 학습
    - **Wide 파트:** 원-핫 인코딩된 교차 특성을 직접 최종 출력층과 연결하여 **피처 간의 선형적인 관계**를 빠르게 학습
    - 두 파트의 출력이 `Concatenate` 레이어를 통해 합쳐진 후, 최종 출력 레이어를 거쳐 3개 클래스를 예측
- **모델 튜닝 시도:** `keras-tuner`를 사용해 최적의 하이퍼파라미터를 탐색
    - **튜닝 대상:** Dense 레이어의 뉴런 수(`units_1`, `units_2`), 드롭아웃 비율(`dropout`), L2 규제 강도(`l2`), 학습률(`learning_rate`).
    - **최적의 조합:** 첫 번째 Dense 층 뉴런 수 64개, 두 번째 Dense 층 뉴런 수 16개, 드롭아웃 비율 0.3, L2 규제 강도 0.01, 학습률 0.001로 찾아냄

하이퍼파라미터 및 학습 설정

- **손실 함수:** `sparse_categorical_crossentropy`를 사용

- **옵티마이저:** `adam` **옵티마이저**를 사용했습니다. 최종 학습에서는 튜닝을 통해 얻은 최적 학습률 `0.001`을 적용
- **정규화 및 과적합 방지:**
  - `Dropout(0.5)`**:** 튜닝된 드롭아웃 비율인 0.5를 적용하여 모델의 과적합을 방지
  - **L2 규제 (** `regularizers.l2(0.0001)` **):** 모델의 가중치가 너무 커지지 않도록 패널티를 부여하여 과적합을 억제
  - **조기 종료 (** `EarlyStopping` **):** 검증 손실(`val_loss`)이 **5번** 연속 개선되지 않으면 학습을 중단하고 최적의 가중치를 복원하도록 설정

결과→0.4520