**매장의 세일데이터를 활용한 세일 예측**

작성자: [유성오]  
작성일: [2025-04-18]

# 1. 프로젝트 개요

본 프로젝트는 매장의 세일데이터를 활용하여 세일 여부를 예측하는 인공신경망(ANN)모델을 개발하는 것을 목표로 합니다. 본 모델은 판매 데이터에 기반하여 향후 세일 여부를 예측함으로써 매출 관리 및 마케팅 전략 수립에 도움을 주고자 합니다. 모델은 Jupyter Notebook 환경에서 개발되었으며, 은닉층 2개, 활성화 함수 ReLU, 출력층 회귀 모델로 구성되었습니다.

또한, 모델의 성능을 향상시키기 위해 다음 다섯 가지 옵티마이저를 적용하여 비교 분석 하였습니다.

◆ SDG

◆ Momentum

◆ AdaGrad

◆ RMSProp

◆ Adam

각 옵티마이저에 대해 \*\*정확도(Accuracy)\*\*와 \*\*손실율(Loss)\*\*을 측정하여 엑셀 시트로 정리, 성능 비교를 수행하였습니다. 이를 통해 최적의 하이퍼파라미터 조합을 도출하고, ANN 기반의 세일 예측 모델을 구축하는 것이 본 프로젝트의 핵심입니다.

# 2. 데이터 탐색 및 전처리

모델 학습 전, 데이터에 대한 기초적인 탐색(EDA)을 통해 결측치 및 이상치의 존재 여부를 파악하고, 이를 적절할 방식으로 처리하였습니다.

**2.1 결측치 처리**

* + - * **상품 무게(Item\_Weight)**

일부 데이터에서 상품 무게가 결측되어 있었으며, 해당 값은 동일한 품목 코드(Item\_Identifier)를 갖는 상품들의 평균 무게로 대체하였습니다. 동일한 코드로 평균이 불가능한 경우에는 상품 최대 소비 가격(Item\_MRP)의 평균 값을 활용하여 보완하였습니다.

|  |  |
| --- | --- |
| - 결측치 채우기 전후 상품 무게 분포 | |
|  | |
| - 단계별 결측치 채운 개수 | **- 결측치를 채운 후 상품 무게의 평균과 중앙값** |
|  |  |

* **매장 크기(Outlet\_Size)**

Outlet\_Size는 범주형 변수이며, 일부 결측이 존재하였습니다. 대부분의 결측값은 Outlet\_Type에 따라 특정한 패던을 보였으며, 1단계는 매장 유형 + 위치 유형 조합별 최빈값으로 채웠으며 그래도 채우지 못한 값은 최종적으로 Unknown으로 대체 했습니다. 그 중 대부분의 Unknown은 ‘Grocery Store’ 타입에 해당하는 매장들은 모두 ‘Small’ 사이즈임을 확인하였습니다. 이에 따라 해당 결측값은 ‘Small’로 일괄 처리하였습니다.

|  |  |
| --- | --- |
| * 매장 타입 별 분포 그래프 | * 매장 위치별 분포 그래프 |
|  |  |
| * 매장 위치 & 매장 타입 별 Unknown 개수 그래프 | |
|  | |

**2.2 이상치 처리**

* **Item\_Visibility** 와 같은 일부 변수에서 0값이 다수 확인되었습니다. 도메인 관점에서 실제로 진열이 되었지만 상품 매출이 있는 것은 이상하다 생각하여 같은 상품 + 같은 매장 + 상품 유형 조건에 따라 평균 또는 중앙값으로 0인 값들을 대체 하였습니다.

|  |  |  |
| --- | --- | --- |
| Item\_Visibility 박스플롯 | | 상품 노출도 이상치 확인 그래프 |
|  |  | |
| 전처리 전 후 분포 비교 그래프 | | |
|  | | |

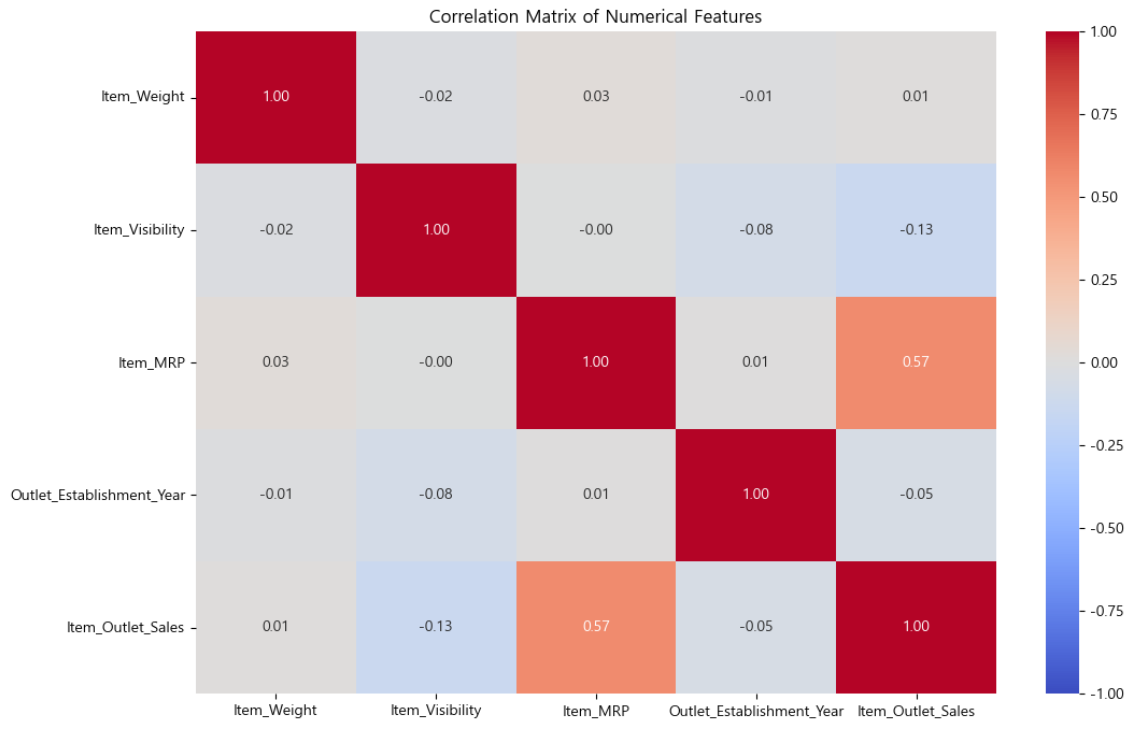
* Item\_Weight 의 경우에는 상관 분석결과 목표 변수와의 연관성이 없기에 이상치 처리 대신 제거를 하였습니다.

# 3. 상관 분석

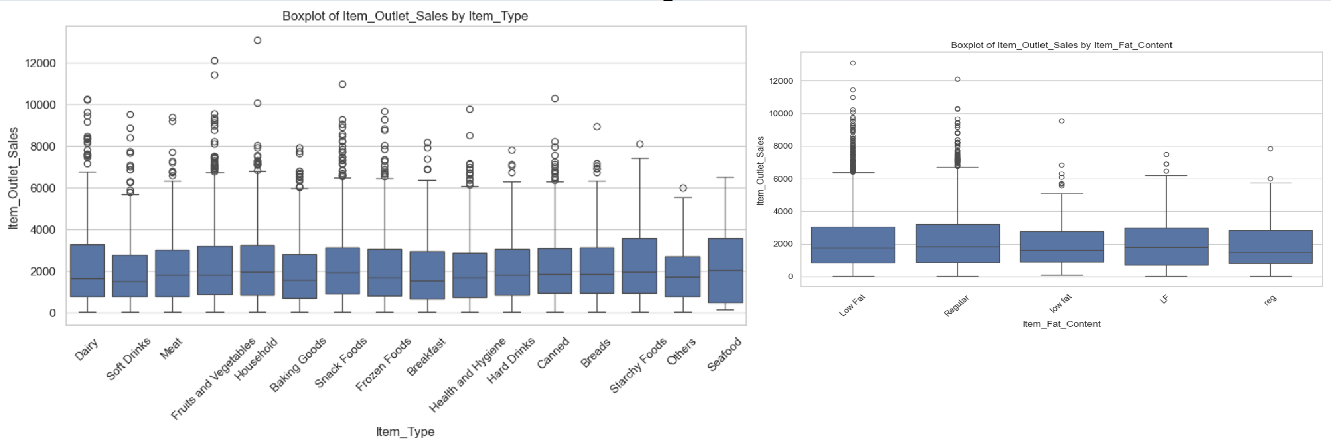
모델링에 들어가기 전에, 데이터 내 변수들 간의 관계를 파악하기 위해 상관 분석을 수행했습니다. 상관 분석은 각 피처들이 목표 변수와 어떻게 연관되어 있는지, 또한 피처들 간의 관계가 어떻게 형성되어 있는지를 이해하는 데 중요한 역할을 합니다. 이를 통해 모델에 포함시킬 변수들을 선정하거나, 다중 공선성 문제를 피하기 위해 상관 관계가 높은 변수들을 제거할 수 있습니다.

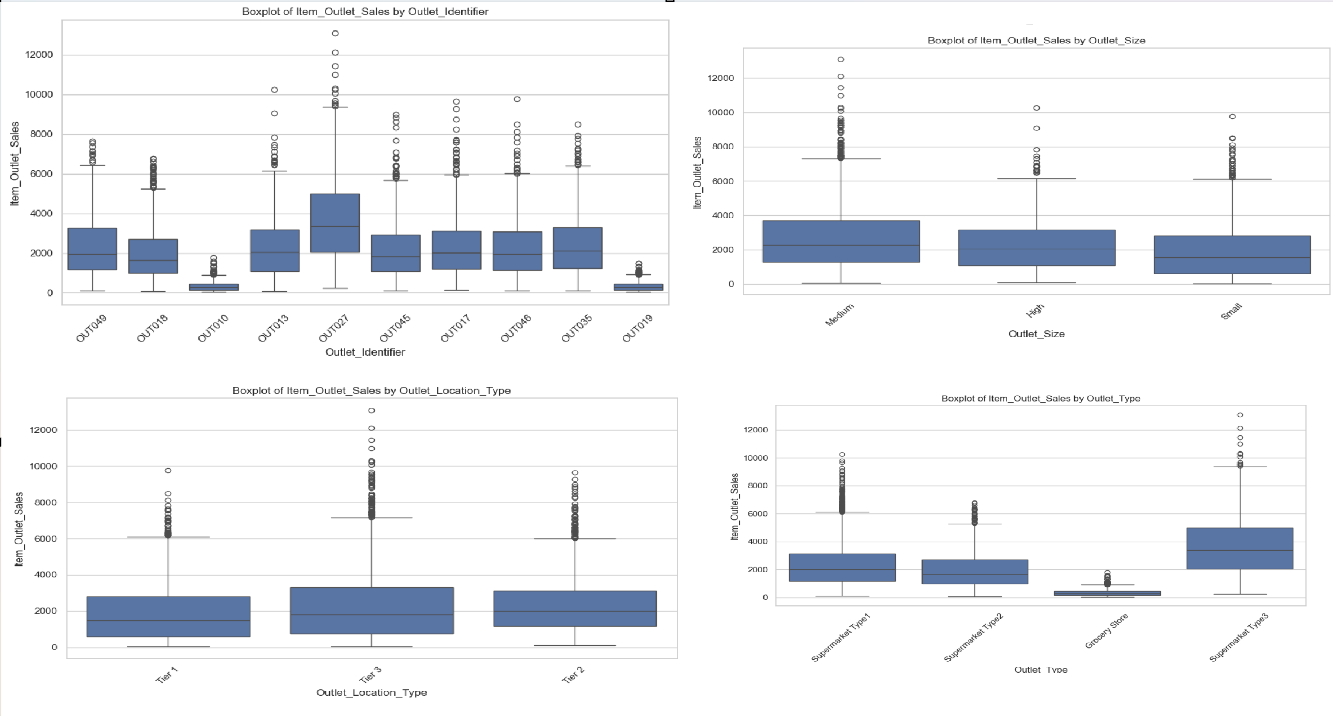
본 분석에서는 변수의 데이터 유형에 따라 서로 다른 통계 기법을 사용하여 관계를 평가했습니다.

* **연속형 변수 간의 관계**를 측정하기 위해 \*\*피어슨 상관 계수(Pearson correlation coefficient)\*\*를 사용하였습니다. 상관 계수 값은 -1에서 1사이의 값을 가지며, 값이 1에 가까울수록 두 변수 간에 강한 양의 상관 관계가 있음을 나타내고, -1에 가까울수록 강상 음의 상관 관계가 있음을 의미합니다. 상관 계수가 0에 가까울수록 변수들 간에 관계가 적음을 나타냅니다.
* **피어슨 상관 계수 그래프**

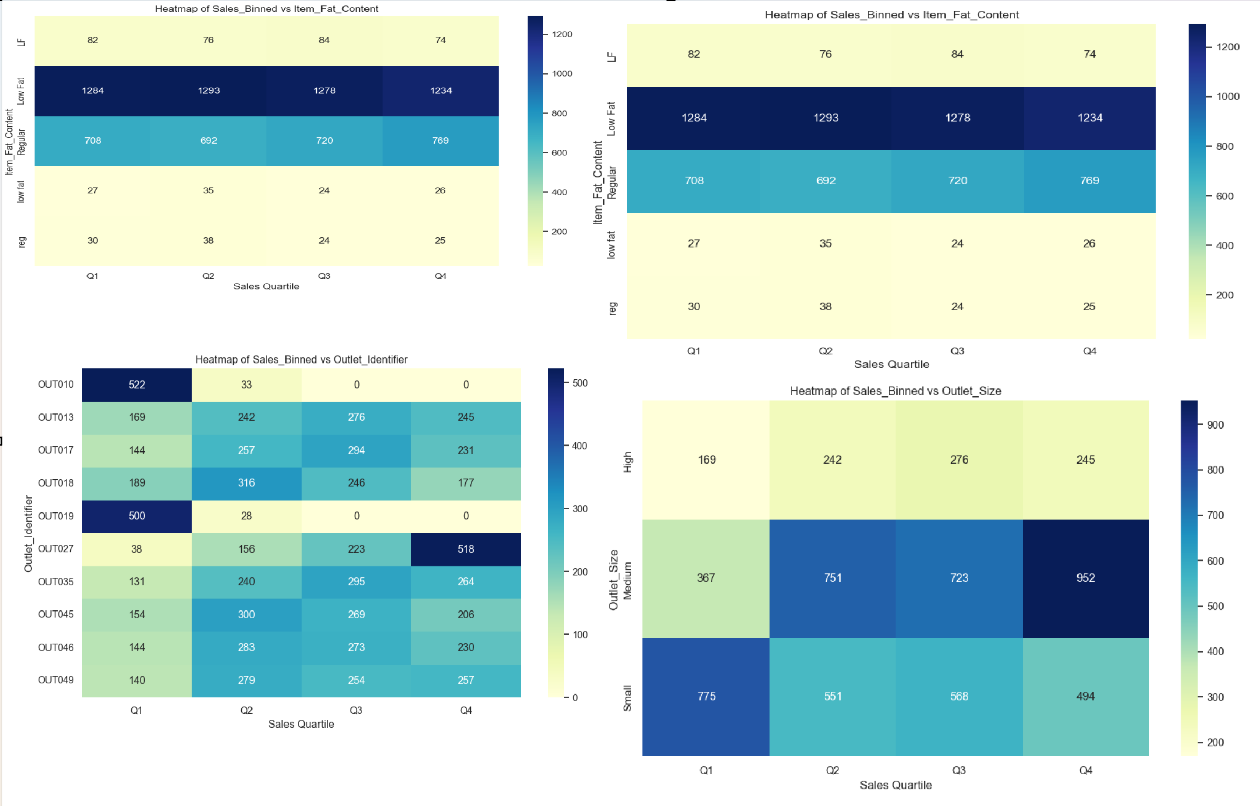


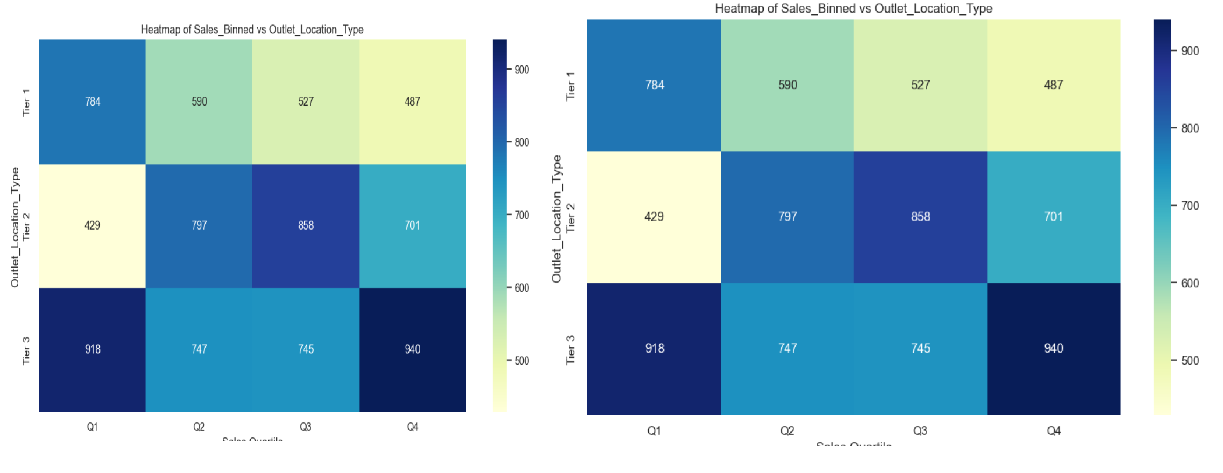
* 범주형 변수와 연속형 목표 변수 간의 관계는 \*\*일원 분산 분석(ANOVA)\*\*를 통해 검정하였습니다. 이는 각 범주가 목표 변수의 평균에 유의미한 영향을 미치는지를 확인 하는데 활용합니다.
* **각 범주형 변수별 boxplot**





* 또한, 범주형 변수들 간의 독립성을 감정하기 위해 \*\*카이제곱 검정(Chi-square test)\*\*을 수행하였습니다. 이를 통해 범주형 변수들 간의 상관 여부를 평가하고, 변수 선택에 참고하였습니다.
* **각 범주형 변수별 교차표 기반 히트맵 시각화**





이러한 다양한 상관 분석 결과를 바탕으로, 모델 입력 변수의 후보를 선정하고, 변수 간 중복성 및 불필요한 변수를 제거하는 데 반영하였습니다.

# 4. 파생변수 생성

원본 데이터의 정보만으로는 충분하지 않을 수 있기 때문에, 모델의 예측 성능을 높이기 위해 **새로운 파생 변수들을 추가로 생성하였습니다.**

파생 변수는 기존 변수 간의 관계를 반영하거나, 특정 패턴을 강조하는 방식으로 설계 하였으며, 주요 변수 간의 상호작용을 더 효과적으로 바녕할 수 있도록 하였습니다. 이 과정은 특징 공학(feature engineering)의 일환으로, 모델이 데이터의 구조를 더 잘 이해하고 학습할 수 있도록 도와줍니다.

**1. 매장 운영 연수**

매장의 설립 연도를 기준으로, 2025년을 기준 연도로 하여 운영 연수(Outlet\_Age)를 파생 변수로 생성하였습니다. 이를 통해 각 매장이 운영된 기간이 모델에 반영되도록 하였습니다.

**2.상품군 분류 단순화**

기존의 상품 품목 개수가 많아 인공 신경망(ANN) 모델에서 과적합(overfitting)이 발생할 가능성이 있었습니다. 이를 방지하기 위해 상품 품목을 다음과 같은 6개 상품군(가공식품, 신선식품, 청량음료,주류,생활용품,기타)으로 그룹화하여 간소화 하였습니다.

**3. 매장별 품목 평균 노출도 기반 상대 노출도**

매장별 상품군의 평균 노출도를 계산한 후, 해당 상품의 노출도를 매장 기준 평균과 비교한 상대 노출도를 새롭게 생성하였습니다. 이 변수는 특정 상품의 매장 내 노출 비중을 파악하는 데 도움을 줍니다.

# 5. 데이터 전처리 완료 데이터

# 

# 6. 모델 설계 및 하이퍼파라미터 튜닝 전략

모델 설계는 **범주형 변수는 Label Encoding을 통해 전처리**하였으며, **수치형 변수는 인공 신경망에 입력하기 전에 StandardScaler를 사용하여 정규화**하였습니다. 이는 학습 속도 향상과 수렴 안정성을 위해 사용하였습니다.

예측 문제는 **회귀 문제**이므로, 기본적인 **인공 신경망(Artificial Neural Network, ANN)** 기반의 회귀 모델을 설계하였습니다.

또한, **하이퍼파라미터 튜닝을 통해 최적의 성능을 도출**하고자 다양한 옵티마이저(SGD, RMSprop, Adam 등)와 **은닉층의 뉴런 수**를 실험적으로 조정해 보았습니다.

모델의 구조는 다음과 같습니다:

**입력층**: 32개의 뉴런, 활성화 함수 'ReLU'  
**은닉층**: 16개의 뉴런, 활성화 함수 'ReLU'  
**출력층**: 1개의 뉴런 (회귀 문제이므로 활성화 함수 없음)

# 7. 하이퍼파라미터 실험 결과

다양한 옵티마이저와 하이퍼파라미터를 실험하여 최적의 조합을 찾았습니다. 각 옵티마이저 및 하이퍼파라미터에 대한 실험 결과를 표로 정리합니다.

|  |  |  |  |
| --- | --- | --- | --- |
| Optimizer | Loss | val\_Loss | R² Score |
| Momentum | 1178834.12 | 1241083.50 | 0.6205 |
| AdaGrad | 1174628.88 | 1244086.88 | 0.6191 |
| SGD | 1194352.50 | 1249653.62 | 0.6144 |
| Adam | 1198483.12 | 1261931.50 | 0.6137 |
| RMSProp | 1199018.62 | 1262615.62 | 0.6131 |

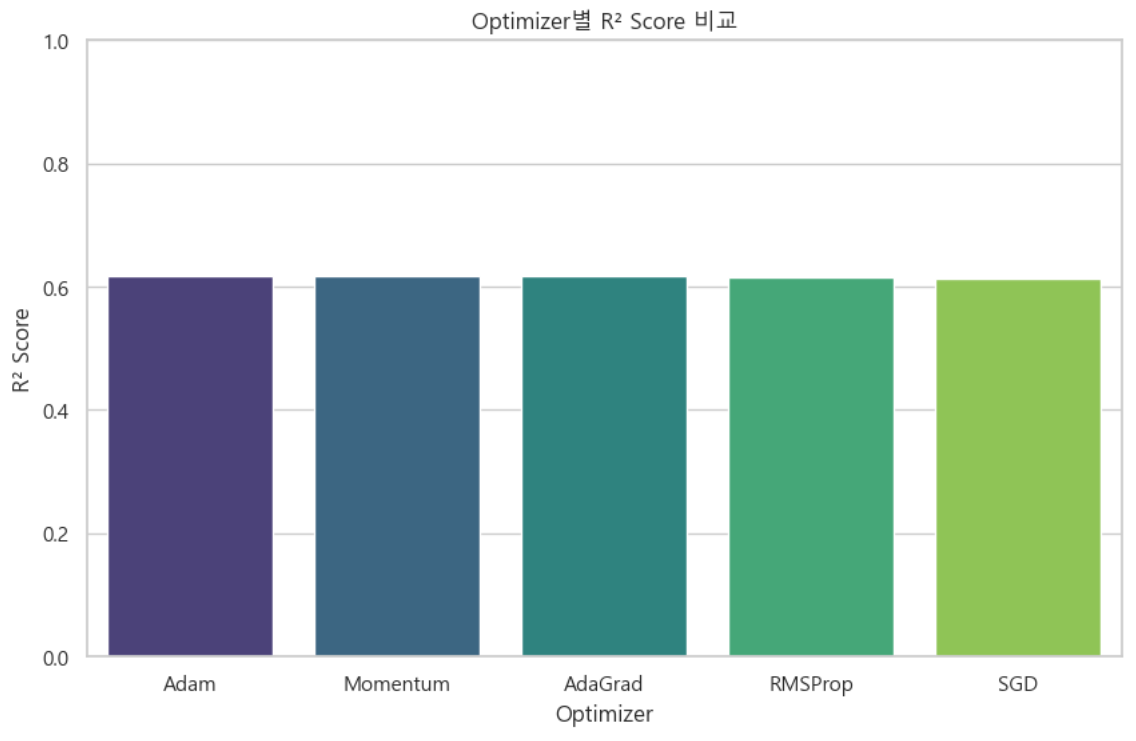
# 8. 예측 성능 비교 및 시각화

모델 성능을 비교하기 위해 다양한 옵티마이저(SGD, RMSprop, Adam 등)를 적용하여 학습을 진행하였으며, 각 모델에 대해 최종 훈련 손실(loss), 검증 손실(val\_loss), R² Score를 계산하여 비교하였습니다.

또한 예측 결과를 시각화하여 모델의 예측 정화도를 직관적으로 확인하였습니다.

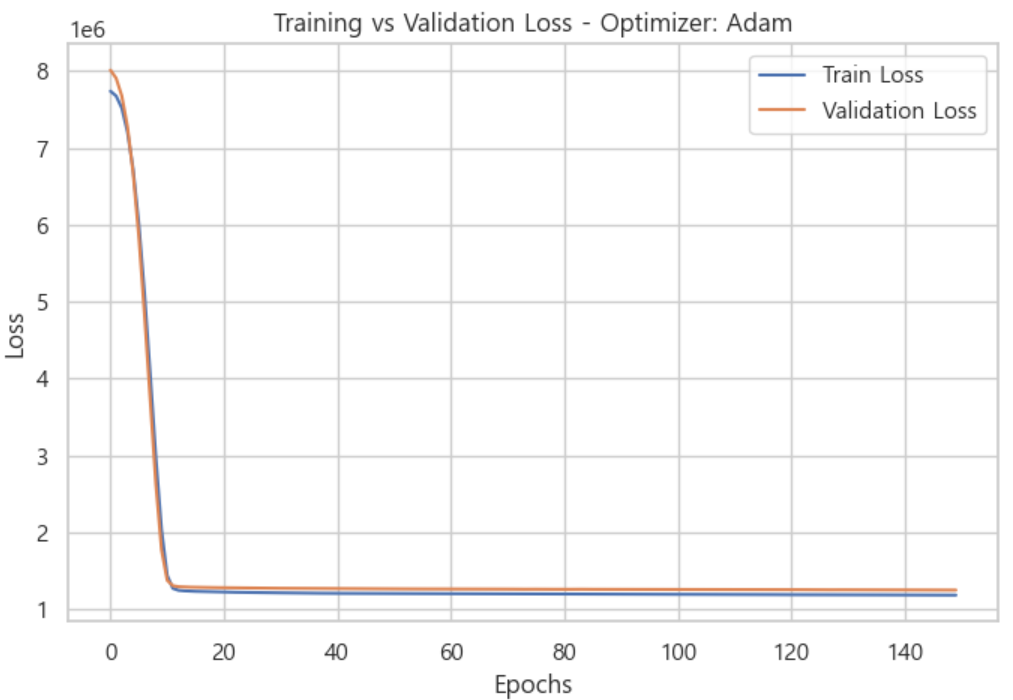
**8\_1 각 Optimizer 별 성능 비교**

옵티마이저별 성능 비교는 아래와 같이 R² Score를 기준으로 정렬된 표 형태로 출력하였으며, 이를 막대 그래프로 시각화하였습니다.



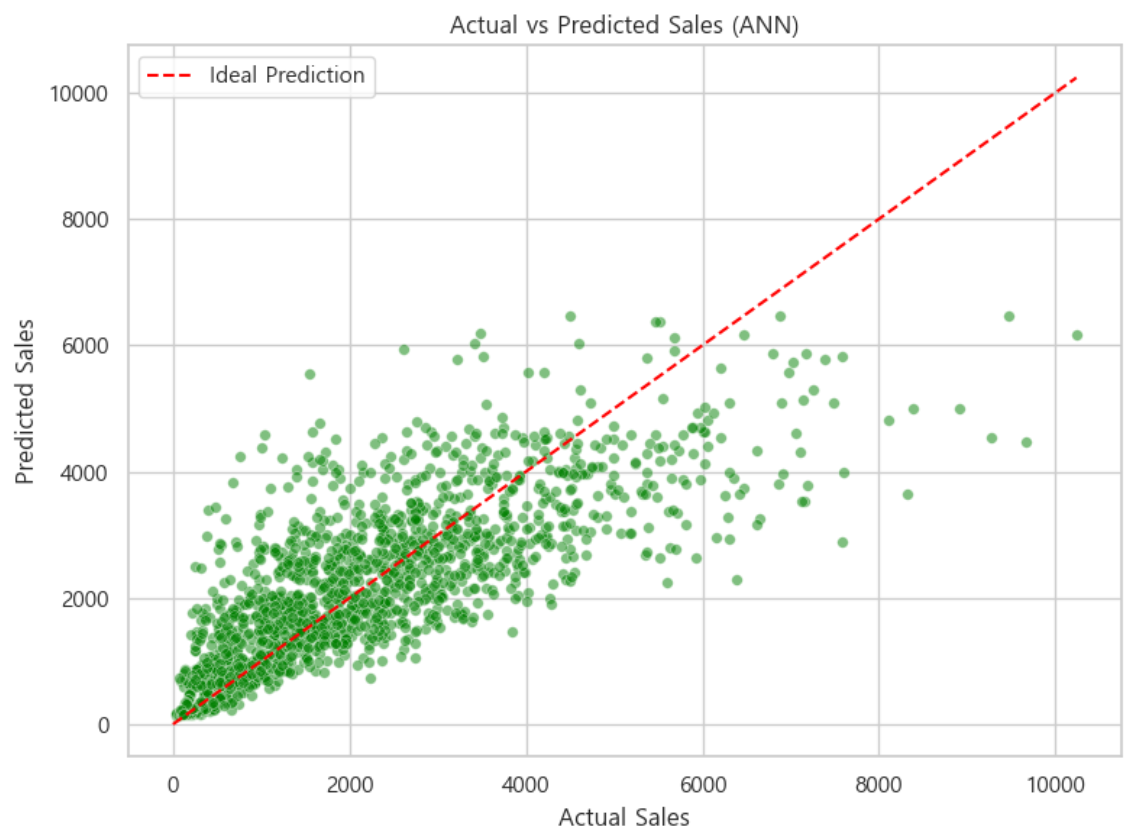
**8-2 최고 성능 모델의 학습 곡선 시각화**

검증 손실(val\_loss)이 가장 낮거나 R² Score가 가장 높은 모델의 학습 이력을 기반으로, 학습 손실과 검증 손실의 변화를 시각화하였습니다.



**8-3 실제값 vs 예측값 산점도**

모델이 예측함 값(y\_pred)과 실제값(y\_test) 사이의 관계를 시각화하여, 모델의 예측 정밀도를 확인하였습니다. 실제값과 예측값이 일치할수록 대각선에 가까운 위치에 점들이 모이게 됩니다.



# 6. 결론 및 개선 방향

본 프로젝트에서는 인공 신경망(ANN)을 활용한 회귀 모델을 설계하고, 다양하 옵티마이저를 적용하여 성능을 비교하였습니다. 모델 학습 시 조기 종료(EarlyStopping)를 도입하고, 학습 및 검증 손실 추이를 모니터링함으로써 과적함을 방지하였습니다.

실험 결과, Momenutum 옵티마이저를 사용한 모델이 가장 우수한 성능을 보였습니다. R² Score 기준으로 Momentum이 가장 높은 예측 정확도(0.6205)를 기록하였으며, AdaGrad, SGD 순으로 뒤를 이었습니다. 손실 값(loss,val\_loss) 측명에서도 Momentum이 안정적인 학습 결과를 보였습니다.

**향후 개선 방향**

**- 하이퍼파라미터 최적화:** 현재는 여러 하이퍼파라미터(batch size, epoch, 은닉층 노드 수 등)에 대해 실험은 수행했으나, 최적화 알고리즘(GridSearch RandomSearch, Bayesian Optimization 등) 을 활용한 체계적인 최적화 과정은 진행하지 않았습니다. 향후 하이퍼파라미터 튜닝을 통해 모델의 예측 성능을 극대화할 수 있습니다.

- **특성 선택 및 엔지니어링 강화**: 파생 변수 외에도 상관관계 분석을 통해 유의미한 특성을 선별하거나 추가 생성하여 예측력을 높일 수 있을거 같습니다.

- **정규화 기법 다양화**: 현재는 StandardScaler를 적용하였으나, 데이터 특성에 따라 MinMaxScaler, RobustScaler 등의 대체 정규화 기법을 사용하여 예측을 높일 수 있을거 같습니다.

이처럼 본 실험에서는 ANN모델 기반의 성능 비교 및 분석을 통해, 데이터에 적합한 옵티마이저와 구조를 탐색하는 기반을 마련하였습니다. 향후 보다 정교한 설계를 통해 예측 정확도를 한층 더 개선할 수 있을 것으로 기대됩니다.

# 7. 사용 도구 및 개발 환경

본 프로젝트에서는 Python을 기반으로 개발되었으며, 인공 신경망(ANN) 모델 설계 및 학습, 데이터 전처리, 성능 평가를 위해 다음과 같은 주요 라이브러리와 도구를 활용하였습니다.

|  |  |
| --- | --- |
| 도구/ 라이브러리 | 사용목적 |
| Python | 전체 모델 개발 및 실험 수행 |
| TensorFlow / Keras | 인공 신경망 모델 설계, 학습, 튜닝 수행 |
| pandas, numpy | 데이터 불러오기, 전처리, 파생변수 생성 |
| scikit-learn | 데이터 분할, 스케일링, 성능 평가 지표 계산 |

모든 실험은 Jupyter Notebook 환경에서 수행되었으며, 모델의 성능 시각화를 위해 matplotlib 및 seaborn을 함께 활용하였습니다.