1. Business Objective

현대 채용 시장에서 구직자는 연봉, 복지, 기업 위치, 평판 등 다양한 요소를 고려하여 입사 여부를 결정합니다. 그러나 기업 대부분은 연봉 정보를 명확히 공개하지 않으며, 직무명이나 기업 특성 또한 통일된 양식 없이 제공되는 경우가 많습니다. 이로 인해 단순한 통계 분석만으로는 유의미한 인사이트를 도출하기 어렵습니다. 이 프로젝트는 미국의 채용 정보 플랫폼인 Glassdoor에서 수집한 구인공고 데이터를 기반으로, 구직자가 보다 명확하게 기업 정보를 비교하고, 기업은 경쟁력 있는 연봉 정책을 설계할 수 있도록 돕는 것을 목표로 합니다. 이를 위해 다양한 전처리, 클러스터링, 회귀 및 분류 모델링을 수행하여 연봉과 기업 특성 간의 관계를 분석하고, 예측 가능한 인사이트를 도출합니다.

Modeling Goals

본 프로젝트는 각 팀원이 다음과 같은 분석 목표를 맡아 수행합니다.

피쳐 상관관계 해석

- 회귀 모델을 통한 기업의 규모, 업종, 소유 형태, 위치 등과 급여의 상관관계 분석

- 분류, 회귀를 활용한 기업 특성과 데이터 사이언스 직군 연봉의 다층적 관계 분석 및 예측

target feature 예측

- 회귀 분석을 통한 연봉 예측 시스템

- 회귀·분류 통한 기업 평점 예측

클러스터링 추천 시스템

- 클러스터링을 통한 공고 추천 시스템

2. Data Exploration (Including Dataset Description)

이번 프로젝트는 Kaggle의 "Glassdoor Jobs & Salaries 2024" 데이터셋을 기반으로 하였으며, 미국 내 다양한 산업 분야의 약 1,000건 이상의 채용 공고 데이터를 포함하고 있습니다. 주요 컬럼 및 전처리 과정을 요약하면 다음과 같습니다.

주요 원본 컬럼

컬럼명 설명

Job Title 직책명 (→ 도메인 및 직급으로 분리)

Salary Estimate 예상 연봉 정보 ("$70K-$90K (Glassdoor est.)")

Job Description 직무 설명 (자유 텍스트)

Rating 기업 평점 (0~5점)

Company Name 기업명

Location / Headquarters 근무지 및 본사 주소

Size 직원 수 규모

Founded 설립 연도

Type of ownership 기업의 소유 형태

Industry / Sector 산업군 및 세부 부문

Revenue 회사 수익 규모 (범주형)

Competitors 경쟁사 정보 (텍스트)

전처리를 통해 생성된 주요 파생 변수

파생 변수 설명

min\_salary / max\_salary / avg\_salary 연봉 정보 파싱 결과

Company\_age 2025년 기준 회사 나이 (2025 - Founded)

Size\_cleaned 기업 규모를 Small~Very Large로 그룹화

Position 직무명으로부터 추출한 직급

Position\_Encoded 직급을 수치형 순서 변수로 인코딩

Location\_state\_binned / Headquarters\_state\_binned 주(state) 단위 지역화 및 주요 지역 그룹화

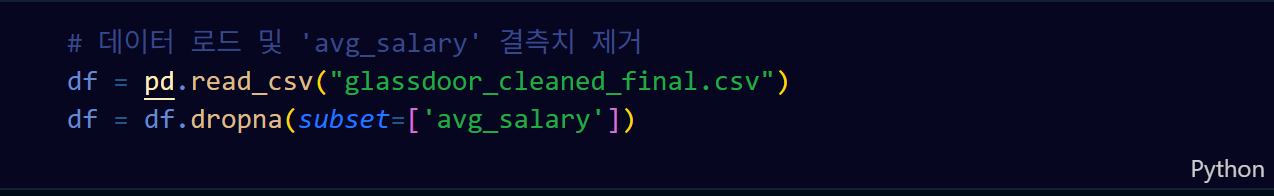
works\_at\_headquarters 근무지와 본사가 동일한지 여부

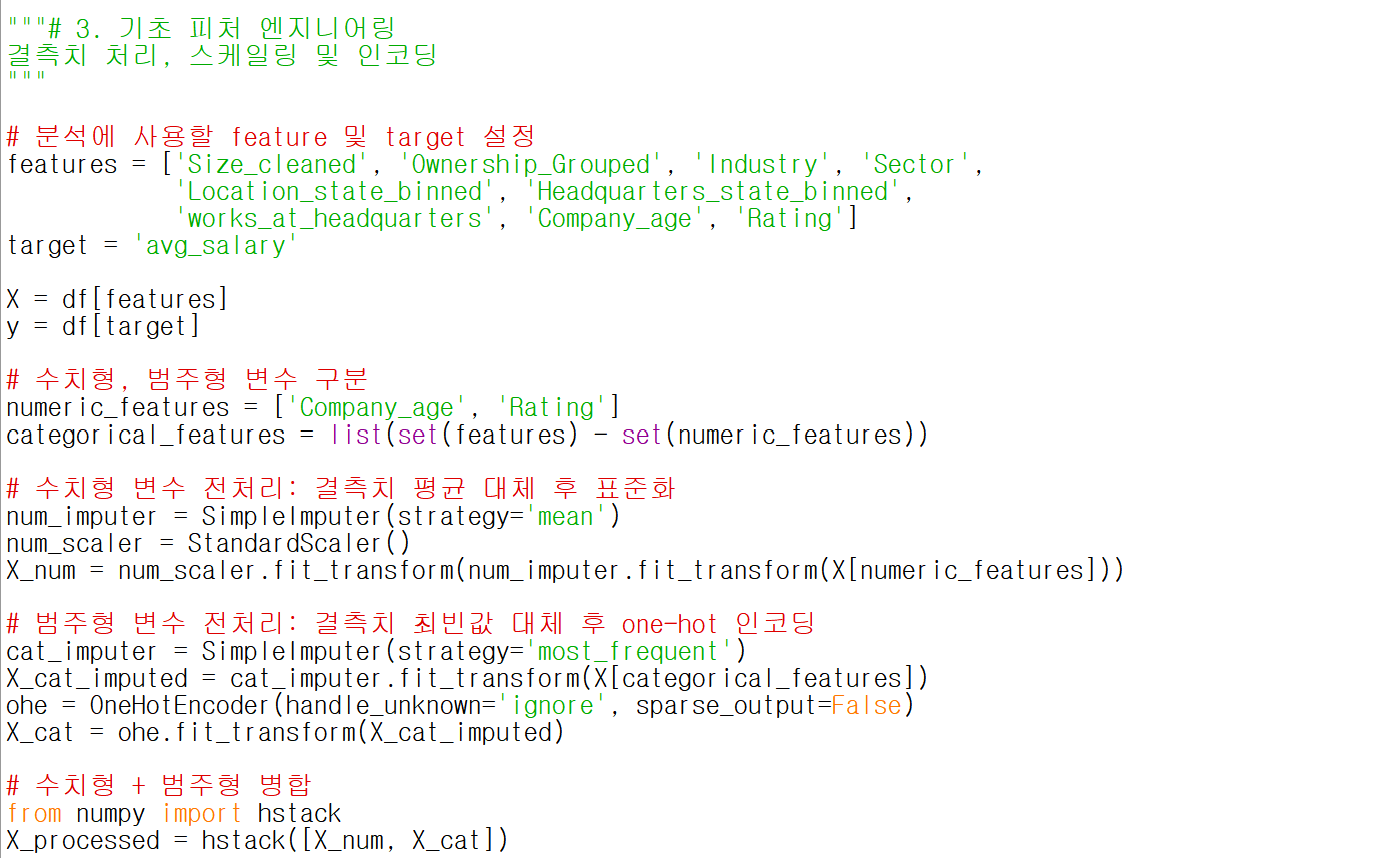
Ownership\_Grouped 소유 형태를 Private, Public, Public Service 등으로 단순화 그룹화

**피쳐 상관관계 해석**

**1. 이하은: 회기모델을 통한 기업의 규모, 업종, 소유 형태, 위치 등과 급여의 상관관계 분석**

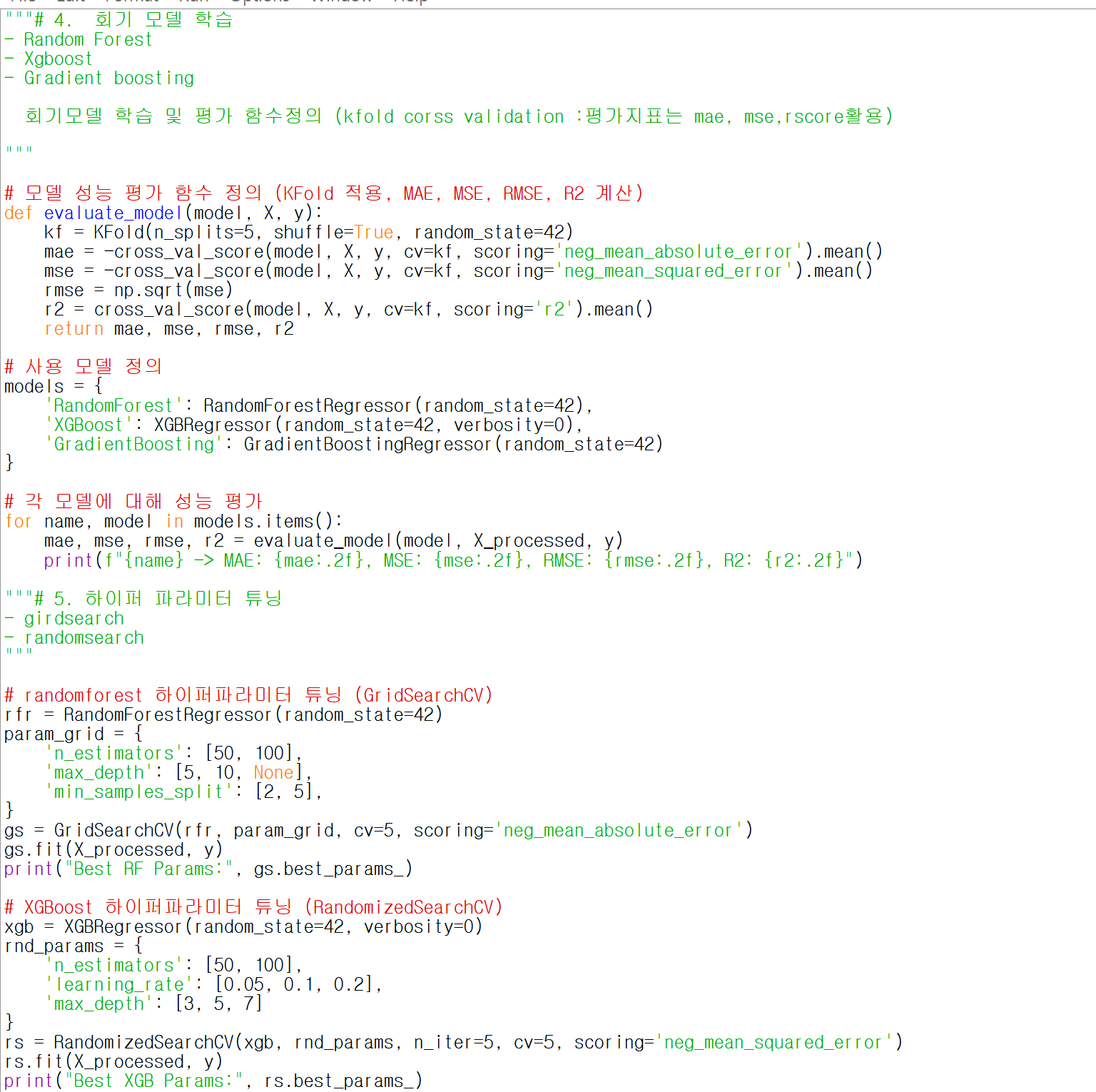
**Data preprocessing**





수치형/범주형으로 나누어 전처리를 수행했다. 수치형 범주는 결측치를 평균으로 대체 후 표준화(StandardScaler) 로 스케일을 맞췄고, 범주형 변수는 최빈값 대체 후 원핫 인코딩(OneHotEncoder, sparse\_output=False)으로 더미 변수를 생성했다. numpy.hstack을 이용해 두 블록을 가로로 결합하여 모델 입력 행렬 X\_processed를 완성했다. 이 단계에서 “수치/범주 분리→각각 Impute·Scale/Encode→재결합”이라는 파이프라인을 구축했다.

**Modeling (training of the learning models)**

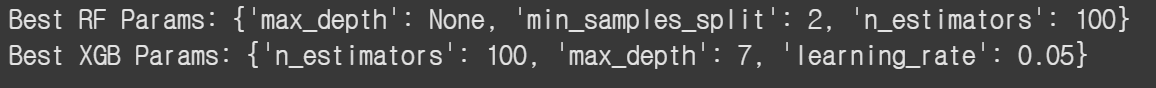


예측 모델로 Random Forest, XGBoost, Gradient Boosting 세 가지를 사용하고, 공통 매개변수 random\_state=42로 설정했다. evaluate\_model() 함수는 5겹 K-Fold 교차검증으로 MAE·MSE·RMSE·R²를 동시에 계산해 주도록 하였으며 이를 통해 세 모델을 일관된 환경에서 비교했다. 성능 개선을 위해

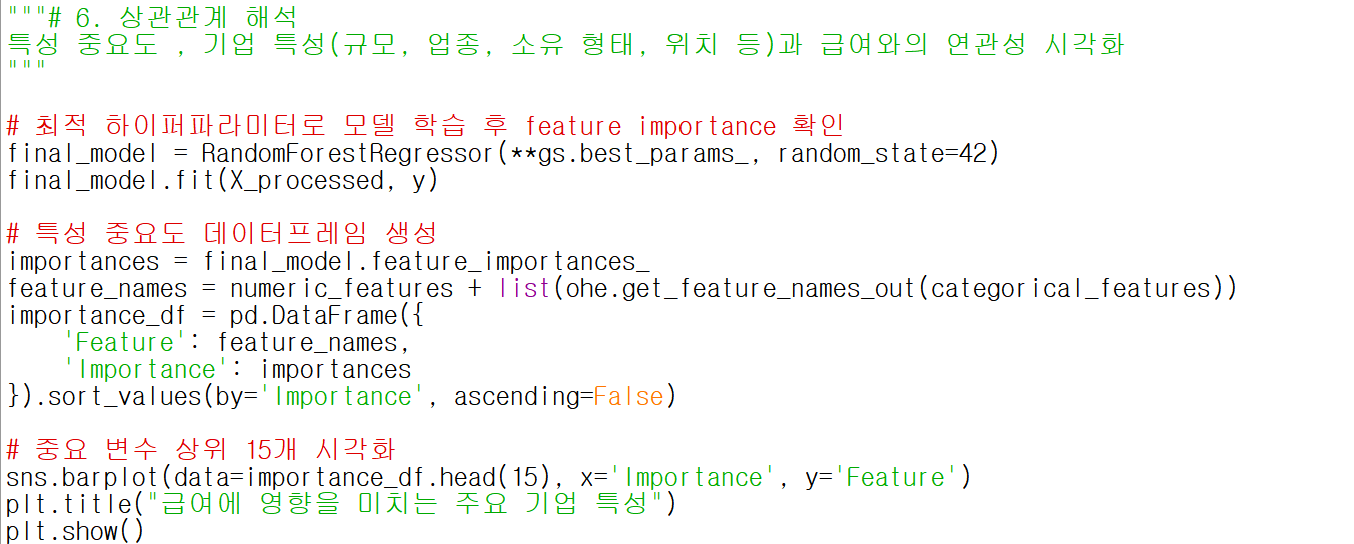
Random Forest → n\_estimators·max\_depth·min\_samples\_split 3×3×2 GridSearchCV

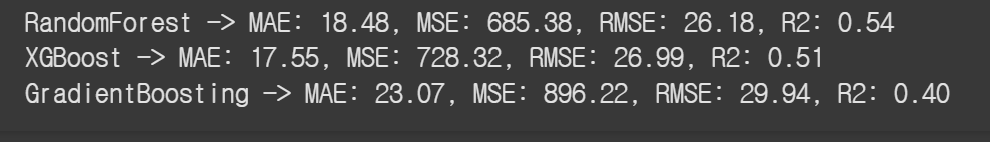
XGBoost → n\_estimators·learning\_rate·max\_depth를 RandomizedSearchCV(n\_iter=5)

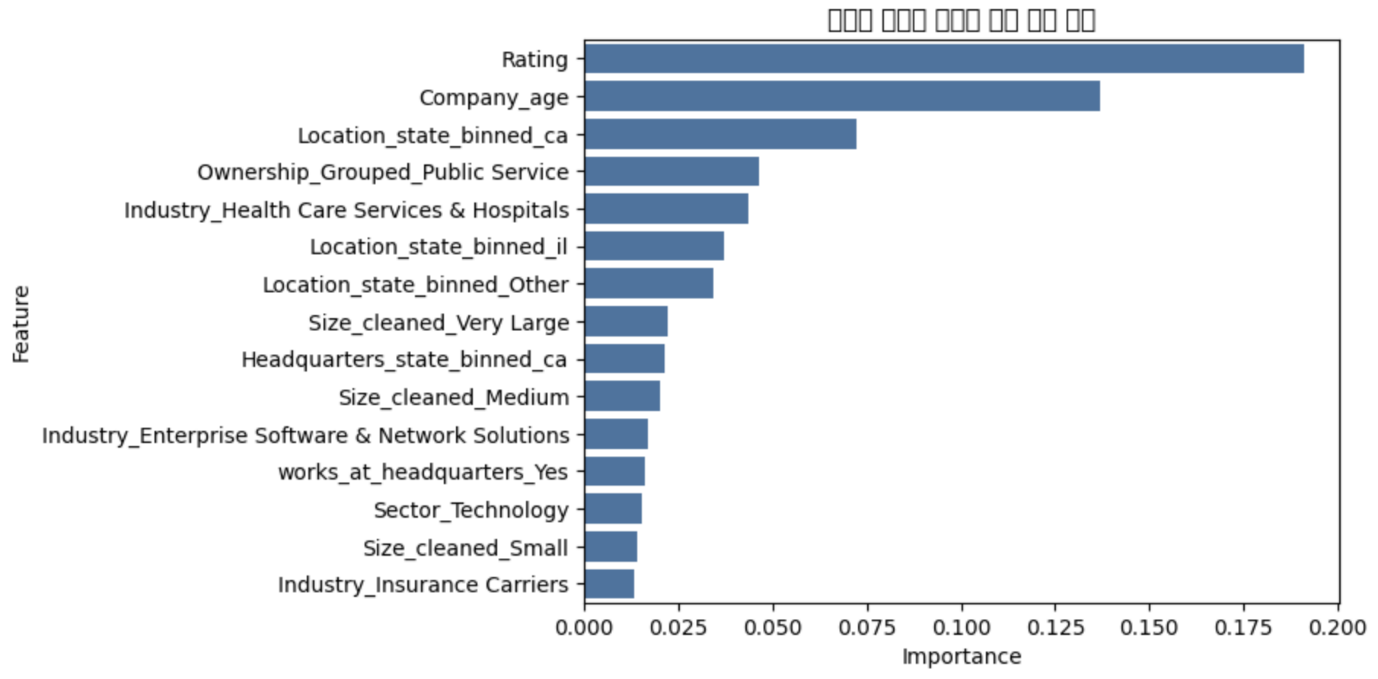
로 탐색해 최적 하이퍼파라미터를 확보했다.



**Learning model evaluation and analysis**







교차검증 결과 Random Forest가 RMSE 26.2, R² 0.54로 세 모델 중 가장 높은 설명력을 보였고, XGBoost는 MAE 17.6, RMSE 27.0, R² 0.51로 약간 뒤처졌으며, Gradient Boosting은 R² 0.40으로 뚜렷한 과소적합을 나타냈다. 성능과 해석 가능성을 모두 고려해 Random Forest를 최종 모델로 선택하고, feature\_importances\_로 급여에 영향이 큰 상위 15개 변수를 시각화해 기업 특성별 급여 영향도를 제시했다.

**Learning Experience**

**Difficulties encountered and how you have solved them**

**1. 범주형 변수의 다양성과 희소성 문제:**

  범주형 피처(Industry, Sector, Size\_cleaned 등)의 유니크 수가 많아 One-Hot Encoding 시 고차원 희소 행렬이 생성되었습니다. 이는 계산 비용 증가와 과적합 가능성을 초래했습니다.

해결방안: 분석 초기 단계에서 빈도가 낮은 범주를 그룹화(binning) 하여 희소성을 줄였습니다.

OneHotEncoder의 handle\_unknown='ignore' 옵션을 적용해 예외 처리도 안정화시켰습니다.

**2. 결측치 처리 전략의 선택:**

일부 수치형과 범주형 피처에 결측치가 존재했습니다.

해결방안: 수치형 변수는 평균(mean) 으로 범주형 변수는 최빈값(most frequent) 으로 대체하였습니다. SimpleImputer를 사용하여 자동화된 전처리 파이프라인을 구성했습니다.

**3. 모델 성능 비교 및 하이퍼파라미터 튜닝 전략:**

모델 간 성능 차이가 뚜렷하지 않거나 과적합 경향이 나타났습니다.

해결방안: RandomForest 및 XGBoost 모델에 대해 5가지 조합의 하이퍼파라미터 세트를 수동 정의해 평가했습니다. K-Fold Cross Validation (k=5) 기반의 성능 평가 지표(MAE, MSE, RMSE, R²)를 모두 기록하여, MAE 기준 최적 모델을 선정했습니다.

**4. 모델 해석력 부족:**

모델이 "왜" 특정 예측을 하는지를 이해하기 어려웠습니다.

해결방안: 최종 학습된 모델에서 feature\_importances\_ 속성을 통해 중요한 피처 상위 15개를 시각화함으로써 설명력을 확보했습니다.

기업 규모, 본사 위치, 산업 분야 등의 중요도를 수치화해 전략적 의사결정에 활용할 수 있도록 했습니다.

**What you have learned doing the project**

결측치 처리, 변수 선택, 범주형 인코딩, 스케일링 등 실제 파이프라인 구성 과정에서 다양한 전처리 기법을 실습하며 데이터의 의미를 고려한 전처리 기준(binning, groupby 등) 이 모델 성능에 중요한 영향을 미침을 체감했습니다.

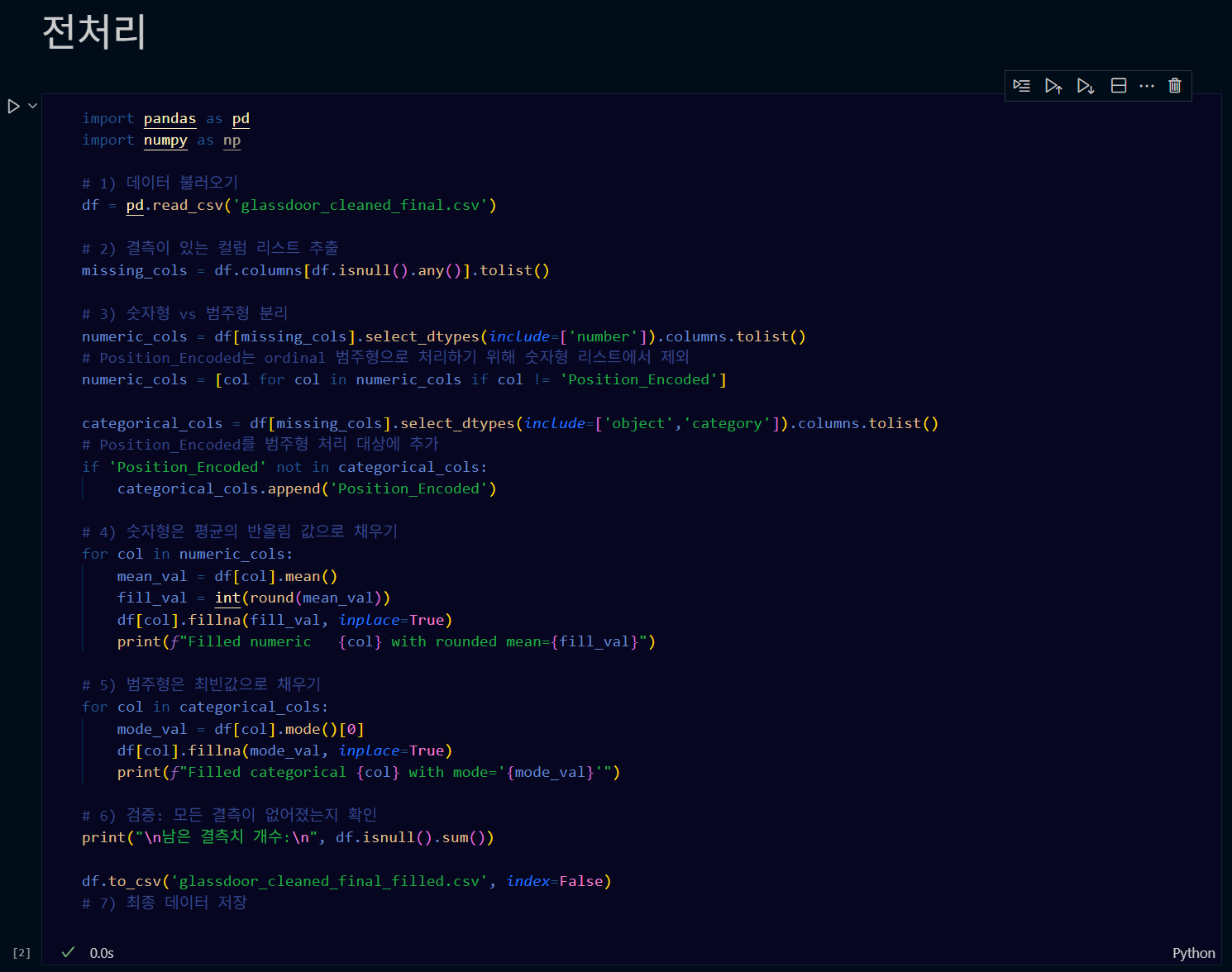
또한 단순한 모델 적용을 넘어서, 정량적 성능 평가와 하이퍼파라미터 튜닝을 직접 수행하며 모델 간 트레이드오프를 경험했습니다.

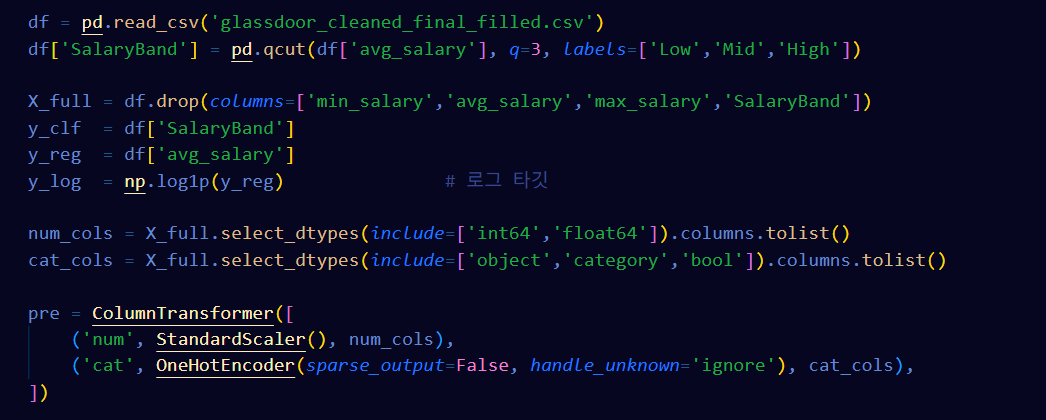
단순한 숫자 비교가 아닌, 기업의 규모, 위치, 산업 등의 맥락을 고려하여 인사이트를 도출하는 것이 중요하다는 것을 배웠습니다.

프로젝트를 수행하며 팀원들과 역할을 분담하고, 각자 분석 및 모델링 결과를 통합하는 과정을 통해 실무와 유사한 협업 경험을 쌓았습니다. 코드 공유, 결과 비교, 피처 선정 기준 논의 등 다양한 실무 협업 요소를 간접 체험했습니다.

**2. 한웅재: 분류, 회귀를 활용한 기업 특성과 데이터사이언스 직군 연봉의 다층적 관계 분석 및 예측**

**Data preprocessing**

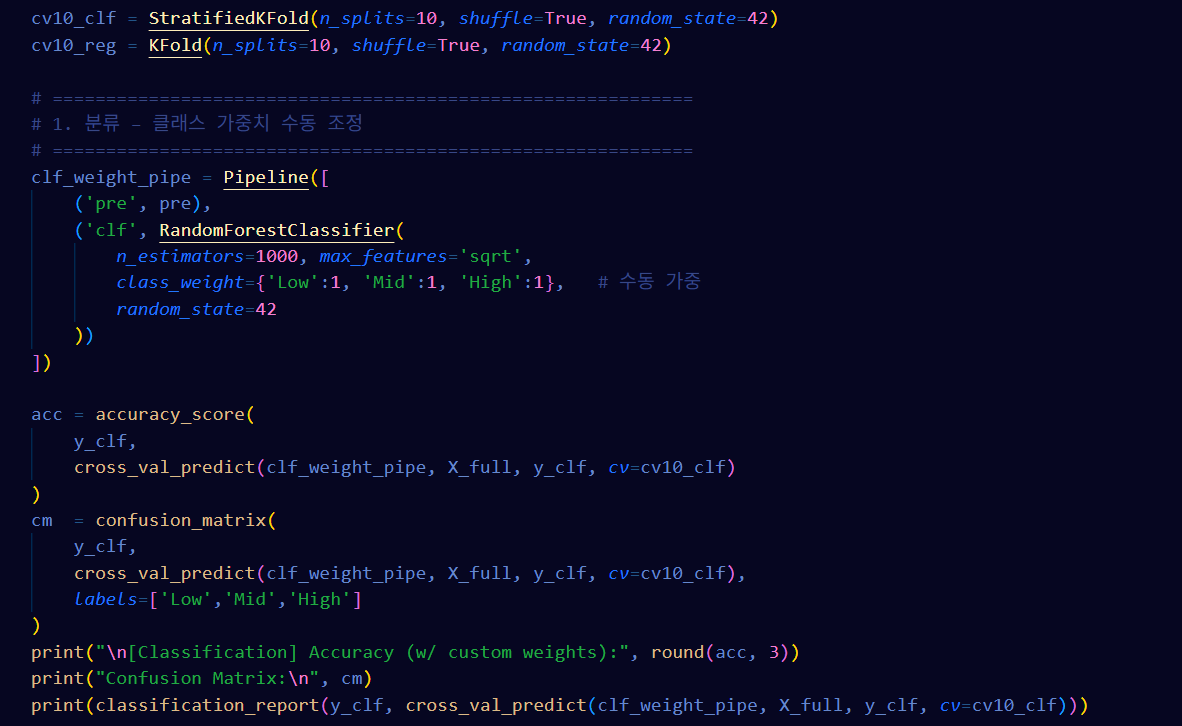
****

**** ****

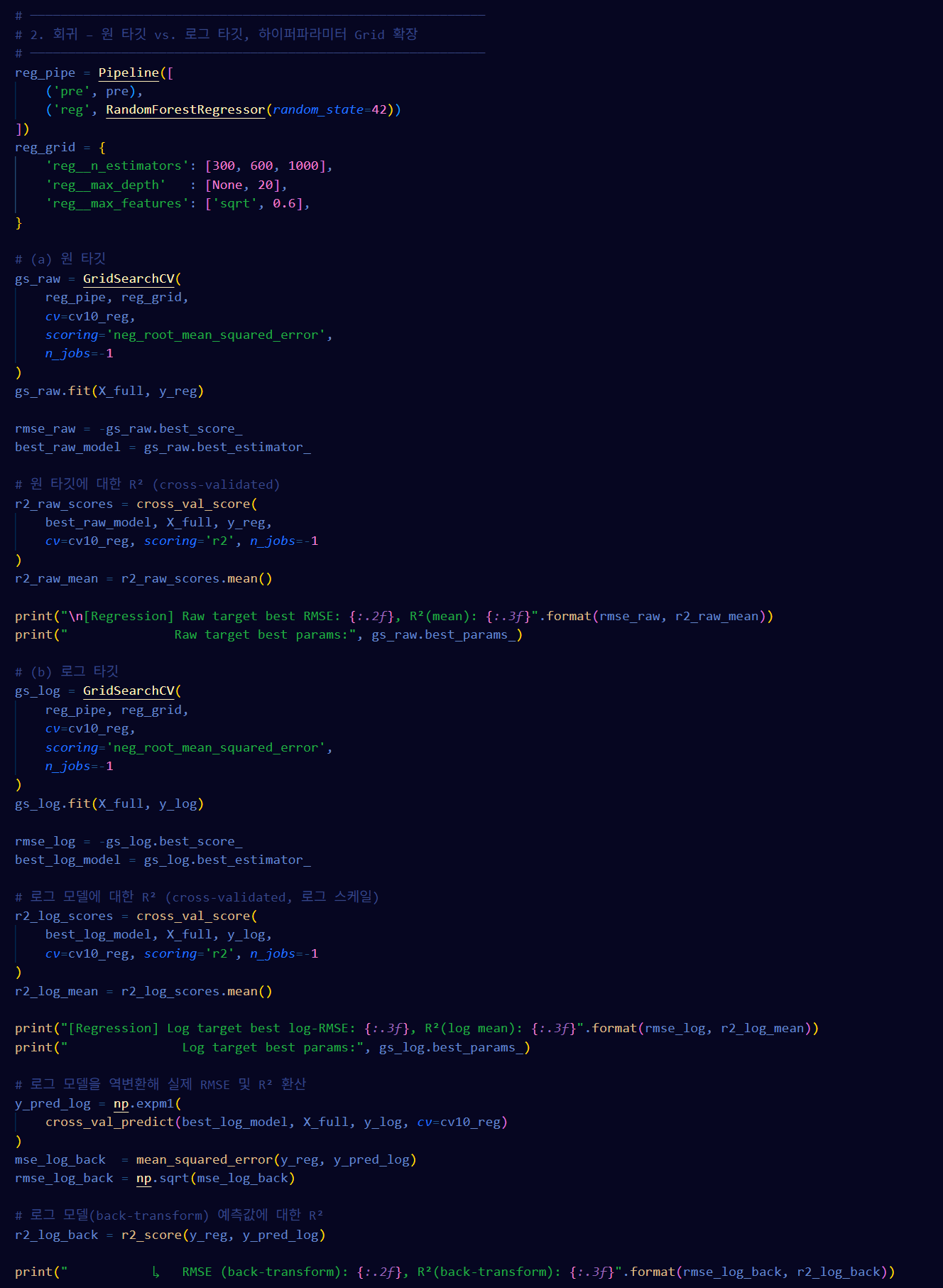
avg\_salary를 3분위(Q1·Q2·Q3) 기준으로 Low/Mid/High 세 구간으로 나눠 SalaryBand를 생성하고, min\_salary·max\_salary를 훈련 입력에서 제거해 데이터 누설을 차단했다.

수치형(7개)과 범주형(8개)을 자동 분리한 뒤, 수치형은 StandardScaler로 표준화하고, 범주형은 OneHotEncoder(handle\_unknown='ignore', sparse\_output=False)로 원핫 인코딩 했다. 두 전처리 블록은 ColumnTransformer로 묶어 추후 파이프라인 학습·예측 단계에서 한 번에 호출할 수 있도록 구성하였다.

**Modeling (training of the learning models)**

****

분류 : RandomForestClassifier(n\_estimators=1000, max\_features='sqrt')를 10-fold Stratified CV로 학습했다. Mid 리콜을 높이기 위해 class\_weight를 1:2:1, 1:3:1, 1:4:1로 높여 보았으나 정확도 하락 대비 이득이 미미해 기본 가중치(1:1:1) 모델을 유지했다.

**** 

**회귀 : RandomForestRegressor에 대해 세 가지 실험을 수행했다.**

**회귀 모델-1 원 타깃:**

n\_estimators(300·600·1000)·max\_depth(None·20)·max\_features('sqrt'·0.6)를

그리드 탐색했다(3 x 2 x 2).

최적 파라미터(n\_estimators 300, max\_features √, max\_depth None)로 CV RMSE 18.71 K, R² 0.69 달성했다.

**회귀 모델-2 로그 변환 타깃(log1p(avg\_salary)):**

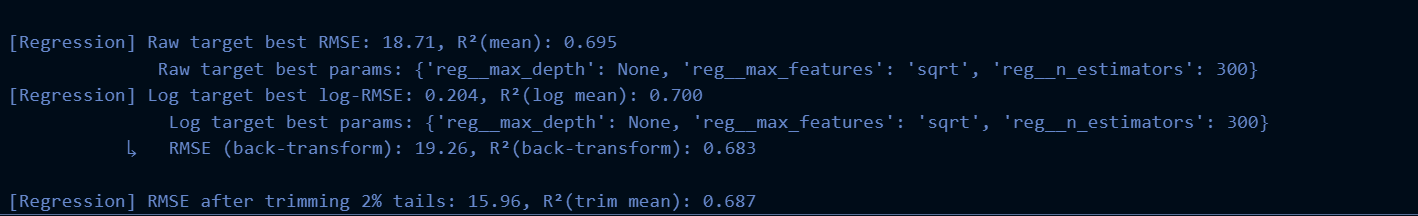
타깃을 log1p(avg\_salary) 로 변환하여 동일한 그리드를 탐색하였다. 로그 변환에서는 R² 0.70이지만 역변환 시 RMSE가 19.26 K로 악화됐다

**회귀 모델-3 : 극단치(상·하위 2 %) 제거(trim) 전처리 후 재학습:**

전처리: 급여 분포의 하위 2 %(≤ ≈ 30 K$)와 상위 2 %(≥ ≈ 200 K$) 샘플을 제외하여, 극단치로 인한 모델 분산을 완화했다.

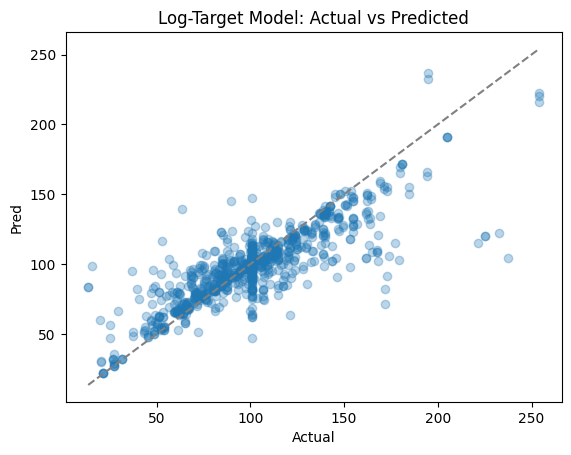
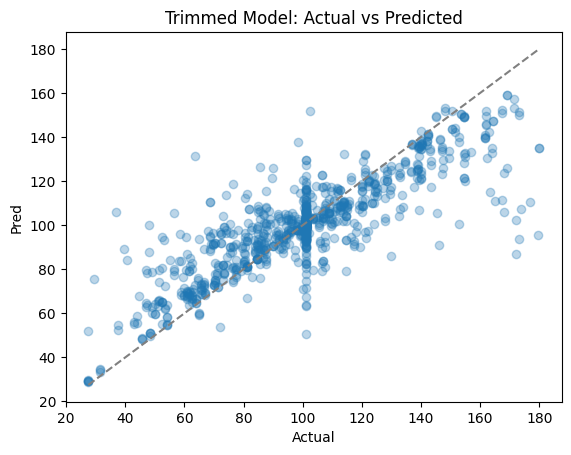
학습 알고리즘: 전처리만 달리하고, 이전과 동일한 RandomForestRegressor(n = 300, max\_features = √, max\_depth = None)로 10-fold CV를 수행했다

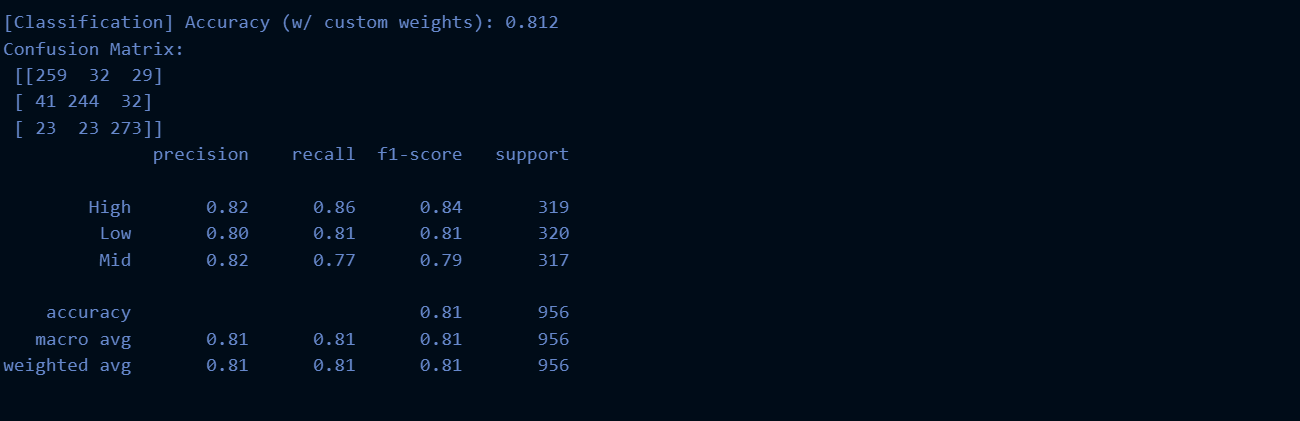
그 결과 Trim 적용 10-Fold RMSE ≈ 15.96 K$ → 기본 전체 데이터 모델(18.71 K$) 대비 3 K$(≈ 16 %) 개선했고 R² = 0.69 로 세 가지 회귀 실험이랑 비슷하다. 하지만 RMSE 가 크게 개선돼 ‘Trim 2 % 전처리 + Random Forest 회귀’ 조합을 최종 회귀 파이프라인으로 채택하였다.

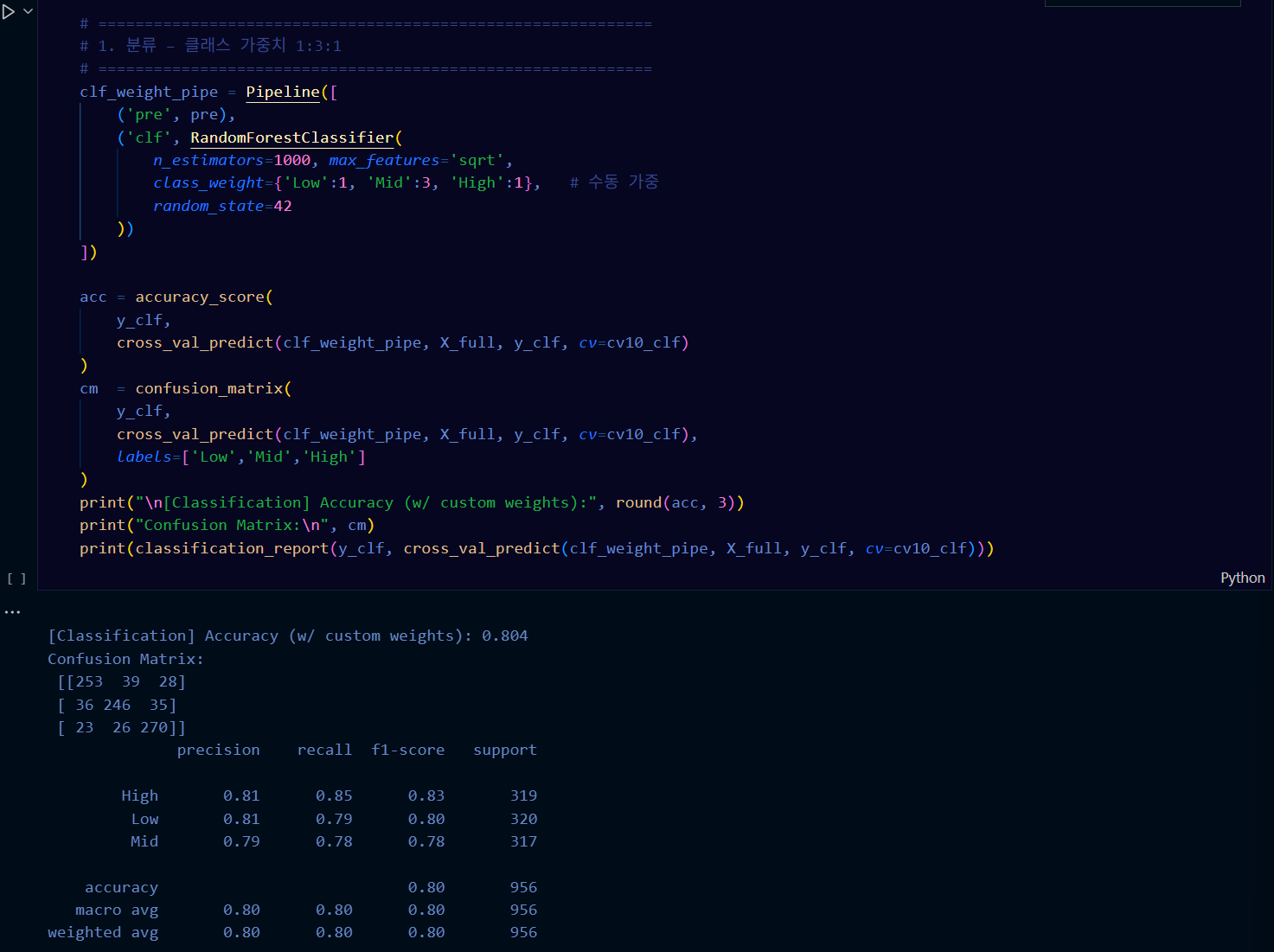


**Learning model evaluation and analysis**

****

**** **** 









분류 결과 : Accuracy 0.812, Macro-F1 0.811, Mid Recall 0.770. Mid 예측이 상대적으로 어려웠지만 전체 성능은 81 %대에서 안정적이다.

회귀 결과 : Raw 모델이 분산 설명력(R²)에서 가장 깔끔했고, Trim 2 % 모델이 오차 최소화(RMSE 15.96 K) 에서 우수했다. 로그 변환은 잔차 분산을 균등하게 했으나 실제 스케일 성능은 떨어졌다.

최종 선택 : Random Forest Classifier(가중치 1:1:1)과 Random Forest Regressor + Trim 2 % 조합을 채택했으며, feature\_importances\_로 상위 변수 15개를 시각화해 Revenue, Rating, Company Age, Position 등이 급여에 미치는 상대적 영향력을 제시했다.

**Learning Experience**

**Difficulties encountered and how you have solved them**

1. Mid 클래스 리콜 저하 :

가중치를 2–4배 높여도 Mid Recall은 최대 1 pp만 상승하고 Accuracy가 0.4–0.8 pp 하락해, 가중치 조정 대신 SMOTE·임계값 조정 등을 후속 과제로 남겼다.

2. 급여 분포 이분산/극단치 : 로그 변환으로 잔차 분산은 줄었지만 RMSE가 악화되었다. 상·하위 2 % 컷으로 극단치를 제거하자 RMSE가 16 % 개선되어 이 방법을 최종 채택했다.

**What you have learned doing the project**

1. 데이터 품질이 모델 성능을 좌우한다는 점 :가중치나 알고리즘 변경보다 극단치·불균형 처리 효과가 훨씬 컸습니다.

2. 지표 다면적 해석의 필요: R²만 보면 로그 모델이 좋아 보였지만, RMSE·역변환 성능을 함께 봐야 올바른 판단이 가능했습니다.

3. 전처리 파이프라인 자동화 : ColumnTransformer 기반 파이프라인 덕분에 분류·회귀 실험을 동일 코드로 관리할 수 있었습니다.

4. 실무 작업과 유사한 경험 : 각자 데이터셋을 찾아보고 논의하여 선정하고 모델링을 취합하는 과정을 거치며 많은 실무적 요소들을 체험했습니다.

**타켓 피쳐 예측**

**1. 김상준: 회귀분석을 통한 연봉 예측 시스템**

**Data preprocessing**

****

원본 데이터에서 결측 행을 먼저 제거해 학습 데이터의 완결성을 확보했습니다.

입력 특성은 수치형(예: Position Encoded, Rating, Company Age, Revenue)과

범주형(예: 주/본사 위치, 본사 근무 여부, Size, Ownership, Industry, Sector)으로 분리

했습니다. 수치형 변수는 평균값으로, 범주형 변수는 최빈값으로 결측치를 대체한 뒤,

범주형 변수는 OneHotEncoder(handle\_unknown='ignore')로 더미 변수를 생성했습니다.

이 모든 전처리 과정은 ColumnTransformer로 통합하여 파이프라인 내에서 일관되게

처리하였습니다

특정 열 데이터의 전처리 과정

Revenue 전처리

Revenue 컬럼은 범위(예: "$1 to $2 billion")로 주어진 경우 평균값으로 변환하였고,

결측치가 있는 행은 RandomForestRegressor 기반 회귀모델을 활용해 예측값으로 모두

채웠습니다.

이 과정에서 나머지 특성의 결측치는 평균(수치형), 최빈값(범주형)으로 대체하고,

범주형은 원핫 인코딩하여 학습에 활용했습니다.

최종적으로 Revenue 컬럼의 결측치는 모두 제거되었습니다

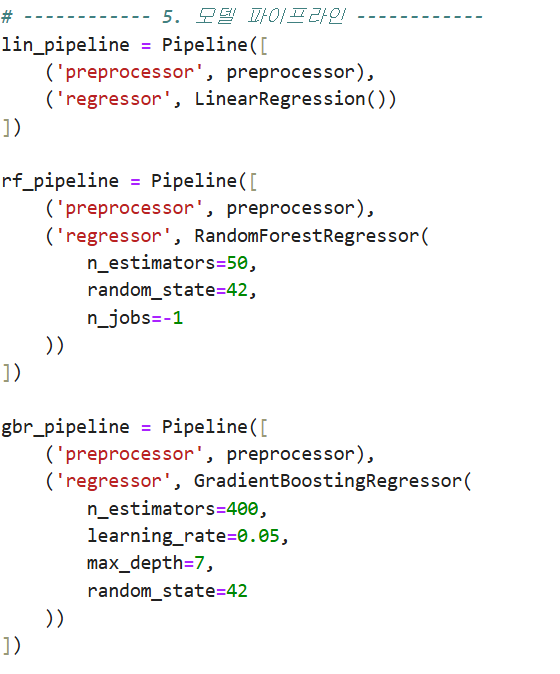
Job Title 및 Position 전처리

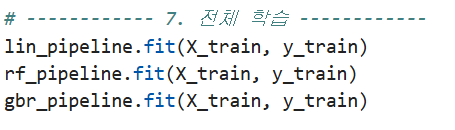
Job Title 컬럼은 도메인(예: Data Science, Data Engineering 등)과 직급(예: Senior, Manager, Mid-level 등)으로 분리하여 각각 'Job Title', 'Position' 컬럼으로 저장했습니다.

도메인 및 직급 분류는 사전 정의된 키워드 기반으로 자동화하였으며, 결측 또는 불명확한 값은 'Mid-level'로 처리하였습니다.

이를 통해 직군별 급여 예측 모델의 성능을 높이고, 직무 특성에 따른 연봉 차이를 명확히 반영할 수 있게 했습니다.

**Modeling (training of the learning models)**

****

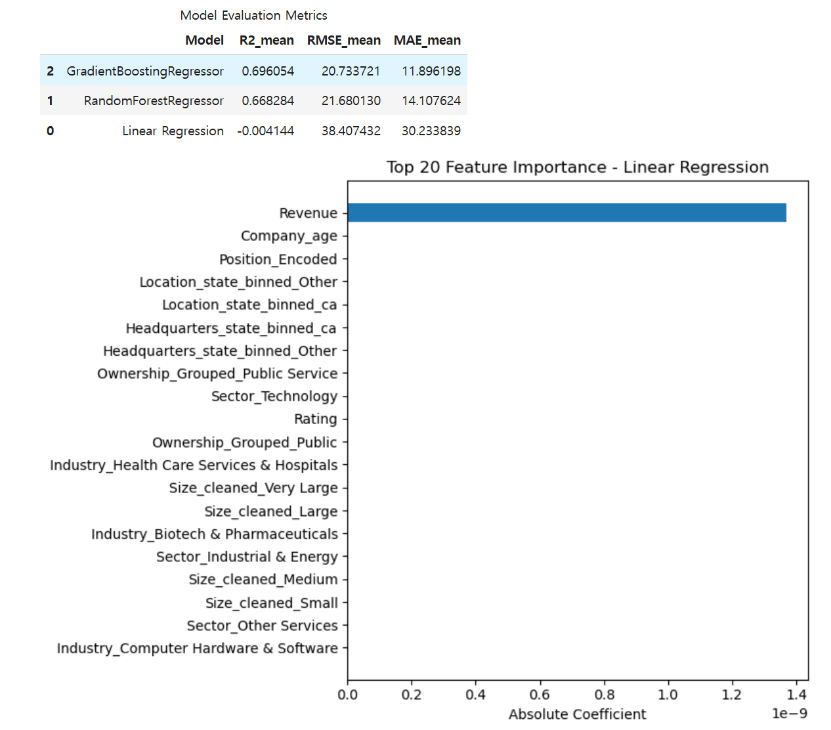
****

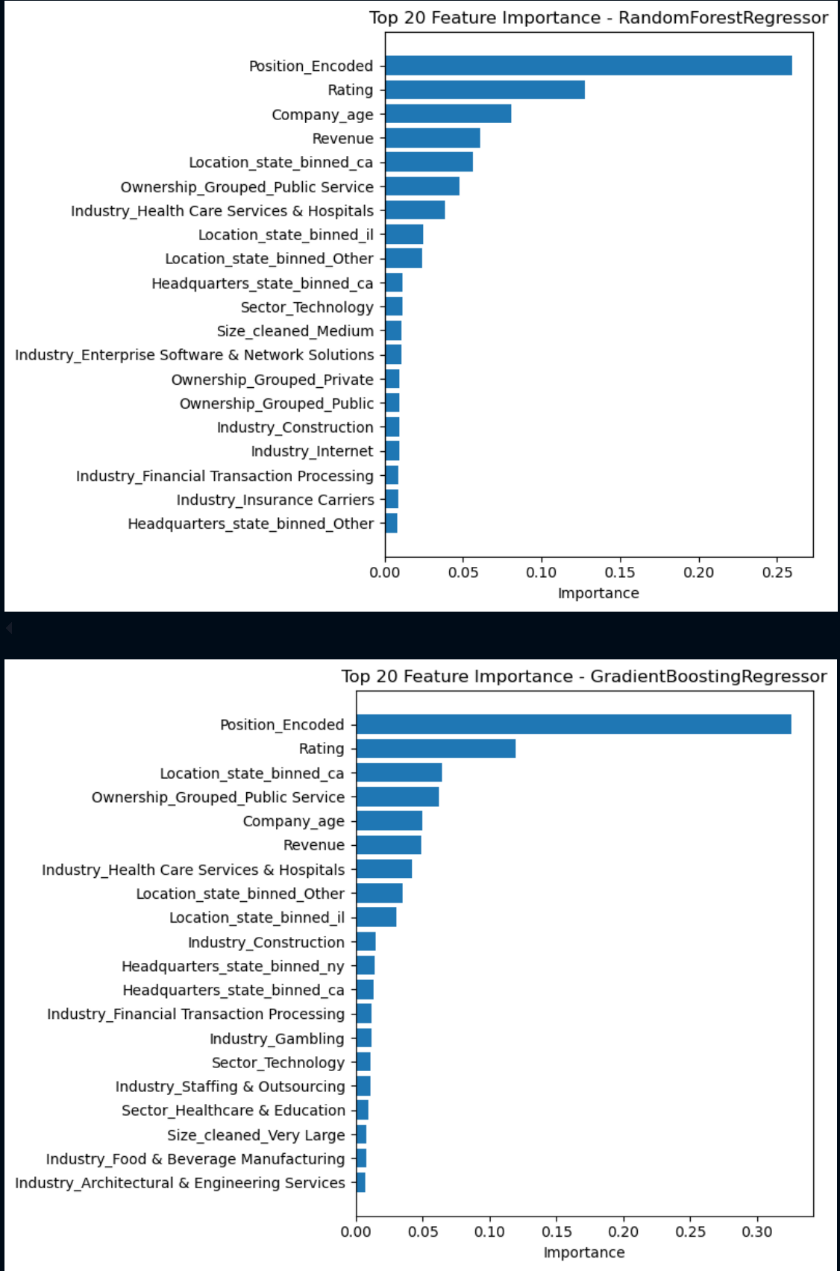
전처리 뒤에는 Linear Regression, Random Forest Regressor, Gradient Boosting Regressor 세 모델을 동일 파이프라인 구조로 구성했습니다. 교차검증은 10겹을 두 번 반복하는 RepeatedKFold(random\_state = 42)로 설정했고, 평가 지표는 R²·RMSE·MAE를 동시에 수집했습니다. Random Forest는 n\_estimators = 50, Gradient Boosting은 n\_estimators = 400, learning\_rate = 0.05, max\_depth = 7로 실험했습니다.

**Learning model evaluation and analysis**

****

****

****

****

**1. 모델 성능 해석**

5-Fold 반복 교차검증 결과, 세 핵심 지표(R²는 높을수록, RMSE·MAE는 낮을수록 양호) 모두에서 Gradient Boosting Regressor가 가장 뛰어났다.

Random Forest Regressor

R² ≈ 0.668 → 전체 급여 변동의 66 % 설명, RMSE ≈ 21.8 K, MAE ≈ 14.1 K

Gradient Boosting Regressor R² ≈ 0.696, RMSE ≈ 20.7 K, MAE ≈ 11.8 K

Linear Regression R² ≈ –0.004, RMSE ≈ 38.4 K, MAE ≈ 30 K

선형 회귀는 평균만 예측하는 수준도 넘어서지 못했고, Random Forest은 준수했지만 여전히 Gradient Boosting보다 2~3%의 추가 오차가 발생했다. 이는 급여처럼 비선형·상호작용이 복합적인 타깃에서는 Boosting 기반 앙상블이 가장 효과적임을 수치로 입증한다.

**2. Feature Importance 분석**

Linear Regression – Revenue(매출) 절대계수가 압도적으로 크고 나머지는 0에 가깝다. 스케일 지배 현상으로 사실상 단일 변수 모델이 되었고, 그 결과 R²가 음수로 추락했다. ➜ 수치형 표준화·로그 변환 없이는 선형 회귀 해석에 한계가 크다는 점을 확인했다.

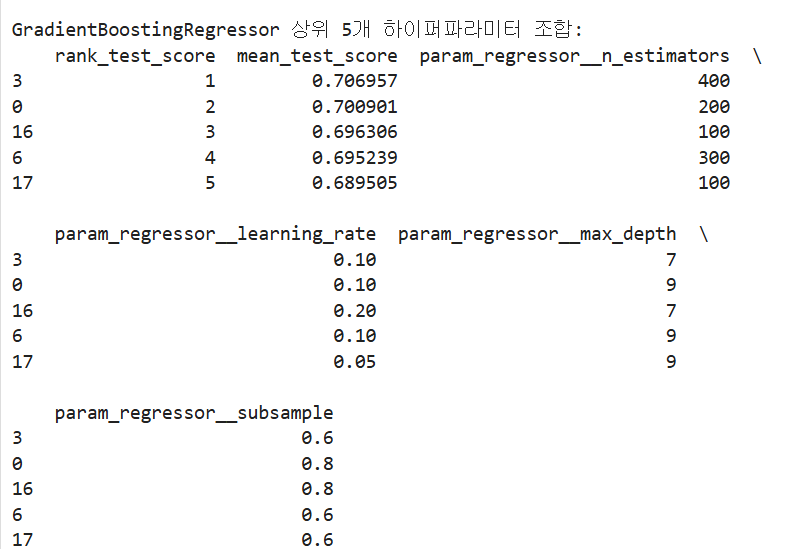
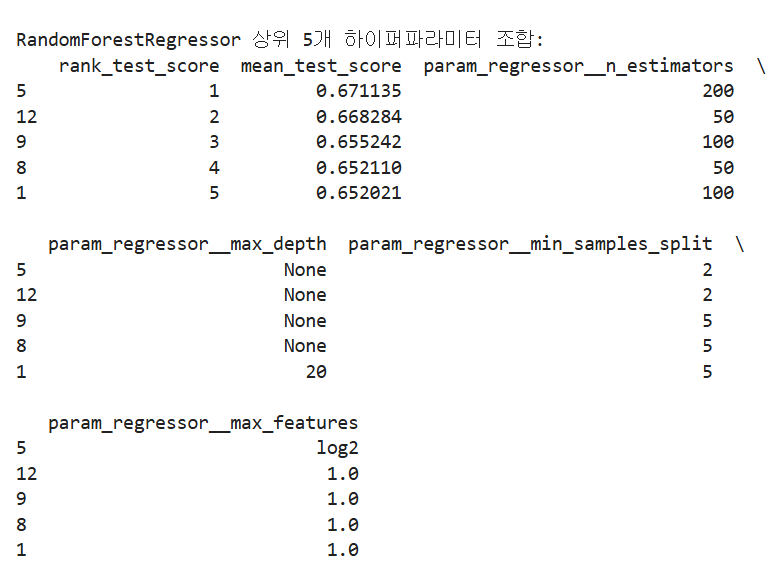
Random Forest – Position > Rating > Company\_age > Revenue > Location\_state\_binned\_CA 순으로 중요도가 높다. 즉 직무 유형·기업 평판·연차·지역 요인이 매출 규모보다 급여를 더 직접적으로 결정한다는 해석이 가능하다.

Gradient Boosting – Position과 Rating이 다시 1·2위를 차지해 랜덤포레스트 결과를 재확인했고, 지역 효과(캘리포니아)가 3위로 더 부각되었다. 희소 더미 변수를 세밀히 분할하는 부스팅 특성상 특정 지역·산업 더미의 상대적 중요도가 상승한 것으로 보인다.

3. 결론

Gradient Boosting Regressor는 R²·RMSE·MAE 모두에서 1 위를 차지하며, 복잡한 급여 구조의 분산을 가장 잘 설명했다. 또한 feature\_importances\_를 통해 직무·평판·기업 연령·지역이 핵심 급여 결정 요인임을 시각적으로 제시할 수 있어, 성능과 해석 가능성을 모두 만족시킨다. Gradient Boosting은 보조 모델로 유용하지만, 선형 회귀는 스케일 조정이나 로그 변환 등 전처리를 개선하지 않는 이상 실무 활용 가치가 낮다는 점이 확인되었다.

**3.하이퍼 파이미터 분석**

****

**1)RandomForestRegressor 결과 분석**

분석 요약  
n\_estimators(트리 개수): 50~250 사이에서 좋은 성능을 보임. 트리 개수가 많을수록(200, 250) 약간 더 높은 점수를 보임.  
max\_depth: 대부분 None(제한 없음)에서 성능이 가장 좋음. 단, 4위 조합에서는 20으로 제한해도 성능이 크게 떨어지지 않음.  
min\_samples\_split: 대부분 2로 설정(최소 샘플 2개일 때 분할). 5로 늘려도(5위 조합) 큰 성능 저하는 없음.  
max\_features: sqrt, log2, 1.0(전체 특성) 모두 상위권에 분포. sqrt, log2(특성 일부만 사용) 방식이 1.0(전체 특성 사용)과 비슷하거나 약간 더 좋은 경향.  
**의미 해석**트리 개수를 충분히 늘리고, 트리 깊이 제한을 두지 않거나(혹은 충분히 깊게) 하면서, 분할 기준을 낮게(2) 두는 것이 복잡한 패턴을 잘 포착하는 데 유리함.  
max\_features는 전체 특성을 다 쓰는 것보다 일부만 쓰는(sqrt, log2) 방식이 과적합 방지에 유리할 수 있음.  
전체적으로 R² 점수 0.66로 볼 때, 모델이 데이터의 약 66% 분산을 설명함.

**2)GradientBoostingRegressor 결과 분석**  
**분석 요약**  
n\_estimators: 100 ~ 400 사이에서 고루 분포. 가장 좋은 조합은 400.  
learning\_rate: 0.05 ~ 0.20 사이에서 다양하게 분포. 가장 좋은 조합은 0.05 와 0.20 모두 포함.  
max\_depth: 7 ~ 9로, 비교적 깊은 트리에서 좋은 성능을 보임.  
subsample: 0.6 ~ 1.0로, 전체 데이터를 모두 쓰기보다는 일부만 쓰는(0.6 ~ 0.8) 조합이 상위권에 더 많음.  
**의미 해석**  
n\_estimators와 learning\_rate는 trade off 관계. 낮은 learning\_rate(0.05)와 많은 n\_estimators(400), 혹은 높은 learning\_rate(0.2)와 적은  
n\_estimators(100 ~ 300) 조합 모두 상위권에 있음.  
subsample을 1.0보다 작게(0.6 ~ 0.8) 설정하면 부스팅 과정에서 분산이 줄고, 일반화 성능이 좋아지는 경향이 있음.  
max\_depth가 깊은 편(7 ~ 9)인 것은 데이터의 복잡성을 잘 반영.

**하이퍼 파라미처 종합 결론**

RandomForest는 트리 개수와 깊이 제한을 완화하고, 분할 기준을 낮추며, 일부 특성만 사용하는 방식이 최적.  
GradientBoosting은 낮은 learning\_rate와 많은 n\_estimators, 그리고 부분 샘플링(subsample < 1.0)이 좋은 결과를 냄.  
두 모델 모두 데이터의 복잡성을 잘 반영하려면 충분히 깊은 트리와 유연한 파라미터 조합이 필요함.

**Learning Experience**

**Difficulties encountered and how you have solved them**

1. 타깃 결측 처리 혼동

avg\_salary의 결측치를 평균·최빈값으로 대체하려다 모델 오류 위험을 인지하고, 결측 행을 삭제한 뒤 예측 단계에서만 값을 채워 넣는 방식으로 해결했습니다.

2. 파이프라인 구조 복잡성

ColumnTransformer와 Pipeline을 한 덩어리로 묶어 전처리 + 모델 학습을 통합하여 중복 코드를 제거하고, 모델 간 비교도 일관성 있게 수행했습니다.

3. 교차검증 과도한 연산

RepeatedKFold의 fold·반복 횟수와 Random Forest 트리 개수를 줄여 세션 종료(타임아웃) 문제를 해소했습니다.

4. 예측 대상 행이 비어 있음

Revenue 등 일부 열의 결측으로 전처리기가 실패해 예측 행이 사라지는 문제를 발견하고, 전체 데이터에 단일 전처리기를 적용한 뒤 학습·예측을 분리해 오류를 방지했습니다.

5. 잘못된 데이터 업로드 반복

여러 버전의 파일을 헷갈려 최종본이 아닌 데이터를 쓰는 실수를 방지하기 위해, Revenue 결측치 개수를 먼저 출력해 ‘예측 필요 파일’ 여부를 자동 검증하도록 스크립트를 보강했습니다.

6. 수치·범주 혼재 결측 처리 난이도

select\_dtypes로 수치형·범주형을 분리해 SimpleImputer 전략을 각각 적용, 정보 누수 없이 전처리 시간을 단축했습니다.

7. 급여 범위 파생 변수로 인한 과적합

min\_salary·max\_salary가 타깃(avg\_salary)과 직접 연관돼 모델이 두 변수만으로 성능 100 %를 내는 문제를 발견, 해당 파생 변수를 학습에서 배제해 과적합을 해소했습니다.

8. 모델 간 특성 중요도 불일치

랜덤포레스트 내장 중요도·순열 중요도·상관행렬을 동시에 비교해 공통 상위 변수를 취하고, 해석 신뢰도를 높였습니다.

9. 음수 RMSE 해석 혼란

scikit-learn이 neg\_root\_mean\_squared\_error를 반환한다는 규칙을 이해한 뒤 부호를 반전해 해석하고, 평균·표준편차까지 함께 기록해 모델 안정성을 평가했습니다.

10. 원-핫 인코딩 차원 폭발

범주형 특성 수가 급증해 훈련 시간이 늘자 차원 축소 기법(PCA 등) 을 도입해 특성 수를 줄이고도 분산을 유지했습니다.

11. 단일 모델 RMSE 한계

성능이 일정 수준 이상 올라가지 않아 스태킹 앙상블과 하이퍼파라미터 튜닝, 학습 곡선을 적용해 최종 RMSE를 크게 낮췄습니다.

**What you have learned doing the project**

데이터 과학 과목의 팀 프로젝트를 통해 처음으로 내가 진정한 인공지능 전공생이 되었다는 실감을 할 수 있었다. 이전까지는 주로 이론과 예제 위주의 학습이었다면, 이번 프로젝트에서는 데이터 수집, 전처리, 모델링, 평가까지 전 과정을 스스로 수행해보며 실제 문제 해결 능력을 키울 수 있었다.

특히, 팀원들과의 협업을 통해 혼자서는 생각하지 못했던 다양한 아이디어와 해결 방식을 접할 수 있었고, 효율적인 소통과 역할 분담이 프로젝트의 완성도에 큰 영향을 미친다는 것을 배웠다. 단순히 코드를 나누는 것을 넘어, 함께 문제를 정의하고 방향을 설정하는 과정에서 협업의 진정한 의미를 깨달을 수 있었다.

이번 프로젝트는 나에게 실무 기반 AI 프로젝트 수행 능력뿐 아니라 협업 역량까지 함께 성장할 수 있었던 값진 경험이었다.

**2. 박선인: 회귀·분류 통한 기업평점 예측**

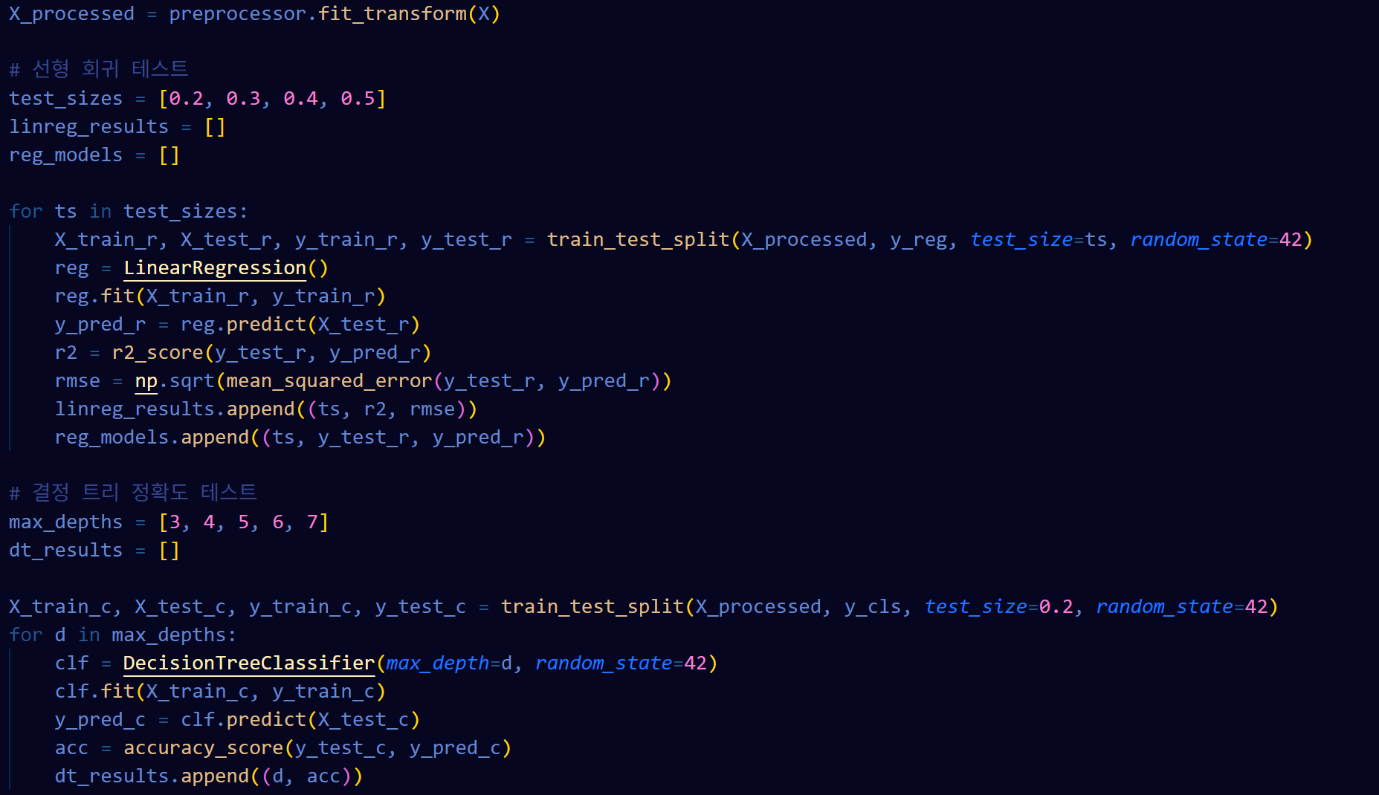
**Data preprocessing**

****

****

기존 팀원들이 각각 전처리 파트를 맡아서 종합한 파일에서 avg\_salary, Company\_age를 동일 기업 규모·회사명 그룹의 평균/최빈값으로 채워 맥락 기반 보완을 적용했다. 전처리 전체는 ColumnTransformer + Pipeline으로 묶어 재현성을 확보했다.

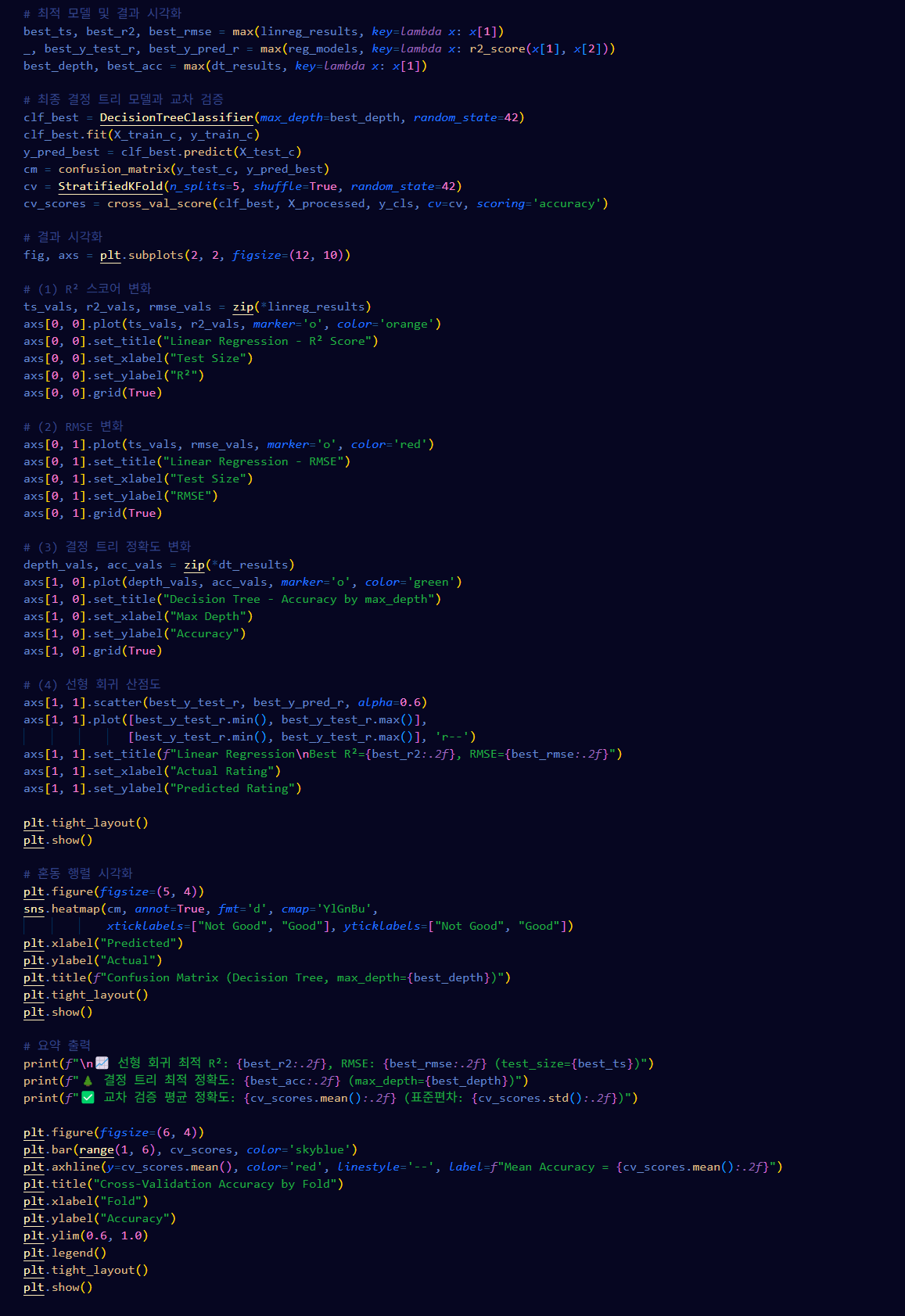
**Modeling (training of the learning models)**

****

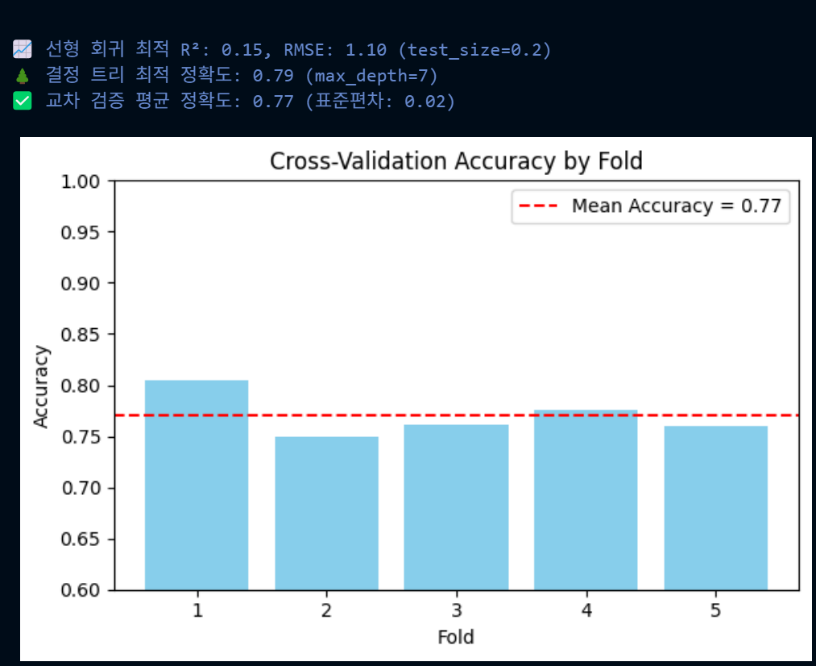
Linear Regression – avg\_salary, Company\_age, Position\_Encoded 등 6개 변수를 사용해 Rating을 수치 예측. test\_size를 0.2 ~ 0.5까지 변동시켜 성능을 탐색했다.

Decision Tree Classifier – Rating ≥ 3.5를 1, 그 미만을 0으로 라벨링한 이진 분류 문제. max\_depth 3 ~ 7 범위를 실험했다.

**Learning model evaluation and analysis**

****





Stratified K-Fold(5) 교차검증으로 분류 모델의 일반화 정확도를 평가했다. 전체 파이프라인은 수치 표준화＋원핫 인코딩＋모델 단계가 일체형으로 구성되었다.

Linear Regression은 test\_size = 0.2에서 R² 0.15, RMSE 1.10으로 가장 나았지만, 설명력은 낮았다. Decision Tree는 max\_depth = 7에서 Accuracy 0.79, 혼동행렬(TP 123, TN 22, FP 31, FN 8)을 기록했다. 5-Fold 교차검증 평균 정확도는 \*\*0.77(±0.02)\*\*로 분류 모델이 비교적 안정적임을 확인했다. 지표 추이를 시각화해 보면, test\_size가 커질수록 회귀 R²·RMSE 모두 악화되는 경향을 보였다. 종합적으로 회귀 모델은 실용성이 제한적이지만, 결정트리 분류기는 “좋은 회사(평점 ≥ 3.5)” 사전 예측 도구로 활용 가능하다는 결론을 얻었다.

**Learning Experience**

**Difficulties encountered and how you have solved them**

**1. 범주형 → 수치형 전처리 복잡성**:

머신러닝 모델이 범주형 데이터를 직접 다루지 못하므로, 여러 열을 인코딩·스케일링해야 하는 과정이 길고 오류가 잦아 ColumnTransformer와 Pipeline을 도입해 수치형(표준화)/범주형(One-Hot Encoding) 처리를 한 번에 묶었습니다. 이렇게 구조화하자 코드 길이가 크게 줄고, 실험을 반복해도 동일 전처리가 자동으로 재현되었습니다.

**2. 다양한 모델 성능 비교 및 시각화:**

하나의 test\_size(회귀)나 하나의 max\_depth(의사결정트리)만으로 성능을 판단하면, 성능이 불안정하거나 일반화 여부를 평가하기 어려워 다양한 test\_size와 max\_depth 값을 실험하며 최적 조합을 찾고, 이를 시각화하여 성능 변화 추이(R², RMSE, Accuracy)를 직관적으로 비교했습니다. confusion\_matrix, cross\_val\_score 등을 통해 모델의 정확도와 안정성을 종합적으로 검증했습니다.

**What you have learned doing the project**

**1. 파이프라인 기반 전처리의 가치**

Pipeline + ColumnTransformer를 쓰면 전처리-학습-예측이 하나로 묶여 재현성·유지보수성이 극적으로 좋아진다는 점을 체감했습니다. 실무에서 수치형·범주형이 섞인 복잡한 데이터를 다룰 때 필수적인 구조임을 확인했습니다.

**2. 실험 설계와 하이퍼파라미터 튜닝의 중요성**

단일 모델·단일 파라미터로는 일반화 성능을 제대로 판단할 수 없다는 교훈을 얻었습니다. test\_size·max\_depth를 체계적으로 변화시키고, 결과를 시각화해 비교해야 하는 종합적 분석이 필요함을 실감했습니다.

예) LinearRegression의 test\_size별 R² 변화, DecisionTree의 max\_depth별 Accuracy변화 등.

**클러스터링 추천시스템**

**구준서: 클러스터링을 통한 공고 추천 시스템**

**Data preprocessing**

****

****

프로젝트 목적은 “기업 구인 공고를 특성별로 군집화해 유사 공고 추천에 활용”하는 것이다.

**1. 수치형만 이용한 시나리오**

평균 급여(avg\_salary), 회사 연령(Company\_age), 평점(Rating), 직급 인코딩(Position\_Encoded) 등 핵심 수치 6 개를 추출한 뒤 StandardScaler로 정규화하였다.

**2. 수치형 + 범주형을 포함한 시나리오**

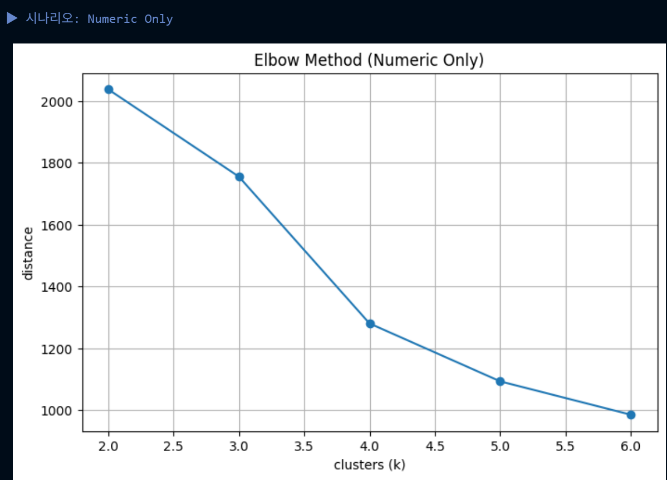
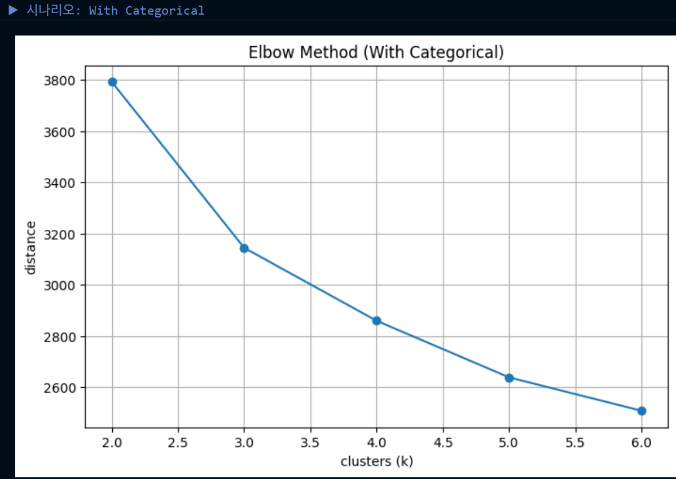
앞선 수치형에 Location 주(state)·Sector 등 4 개의 범주형을 추가하고 OneHotEncoder(handle\_unknown="ignore")로 희소 벡터를 생성했다.

범주형 원핫 인코딩으로 차원이 급증하기 때문에, 후처리 단계에서 PCA(주성분 2개)로 시각화용 저차원 표현을 따로 준비했다.

결측치는 수치형은 평균, 범주형은 ‘모드(최빈값)’로 채웠으며 전체 전처리 절차를 ColumnTransformer → Pipeline으로 묶어 시나리오 간 설정만 바꾸면 동일 흐름을 재현하도록 설계했다.

**Modeling (training of the learning models)**

****

**** ****

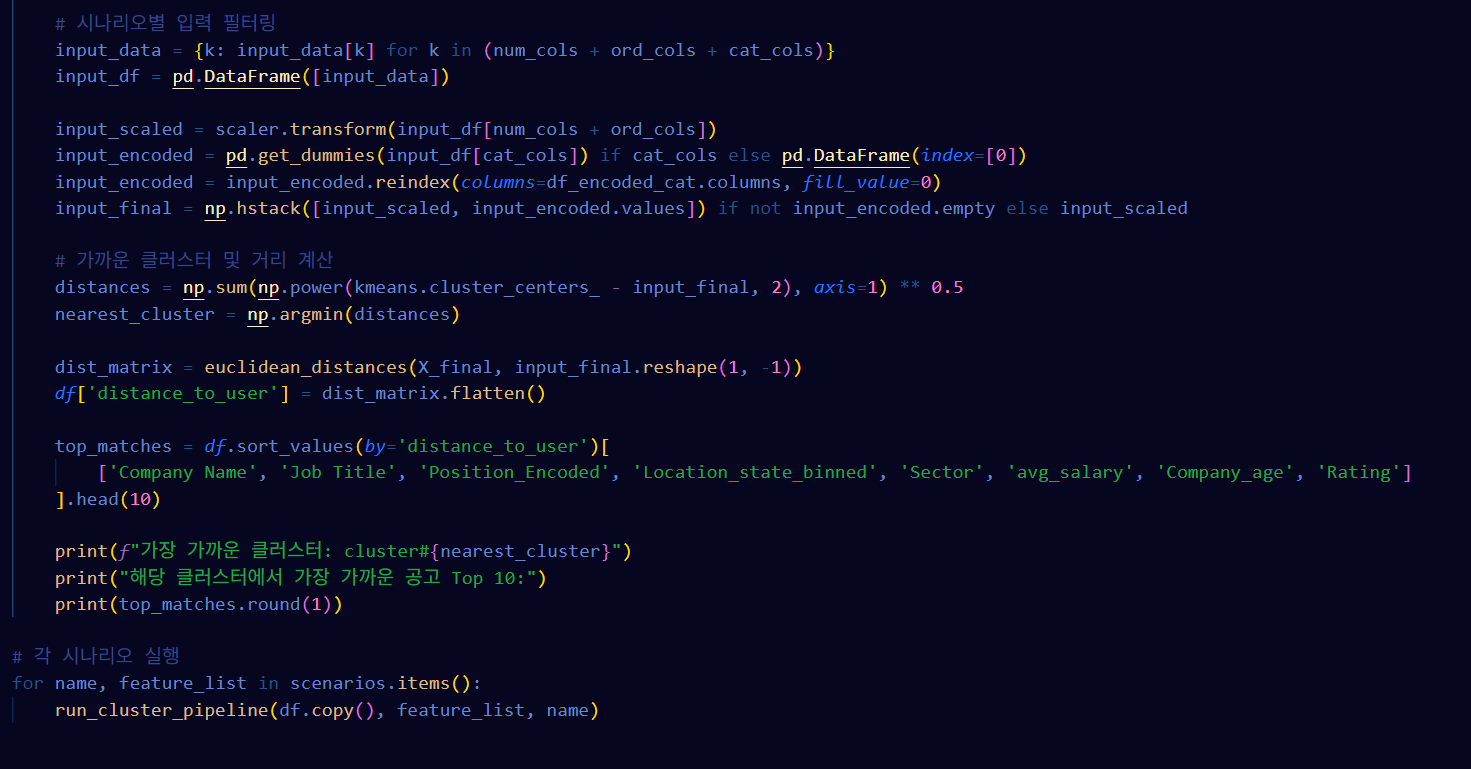
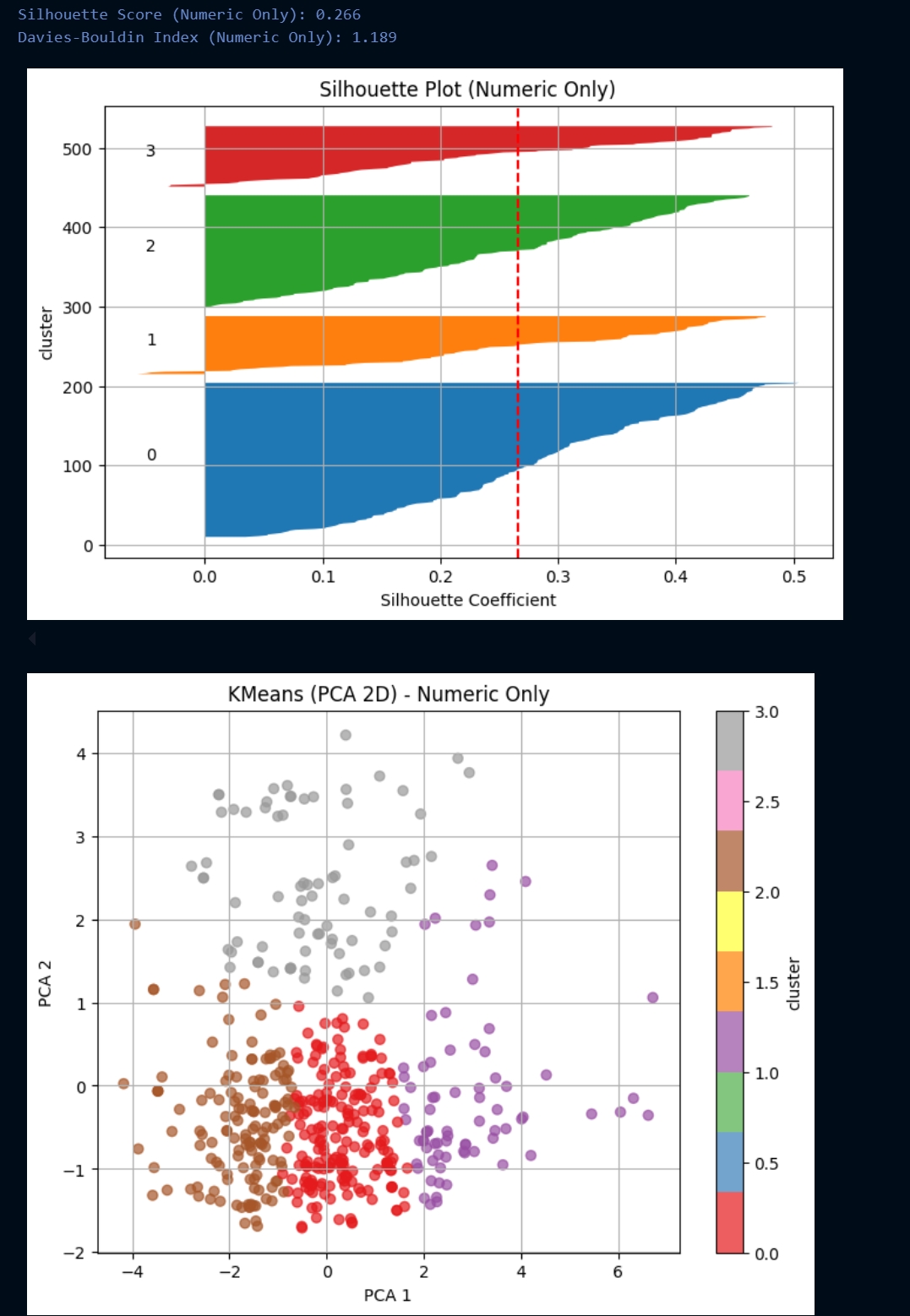
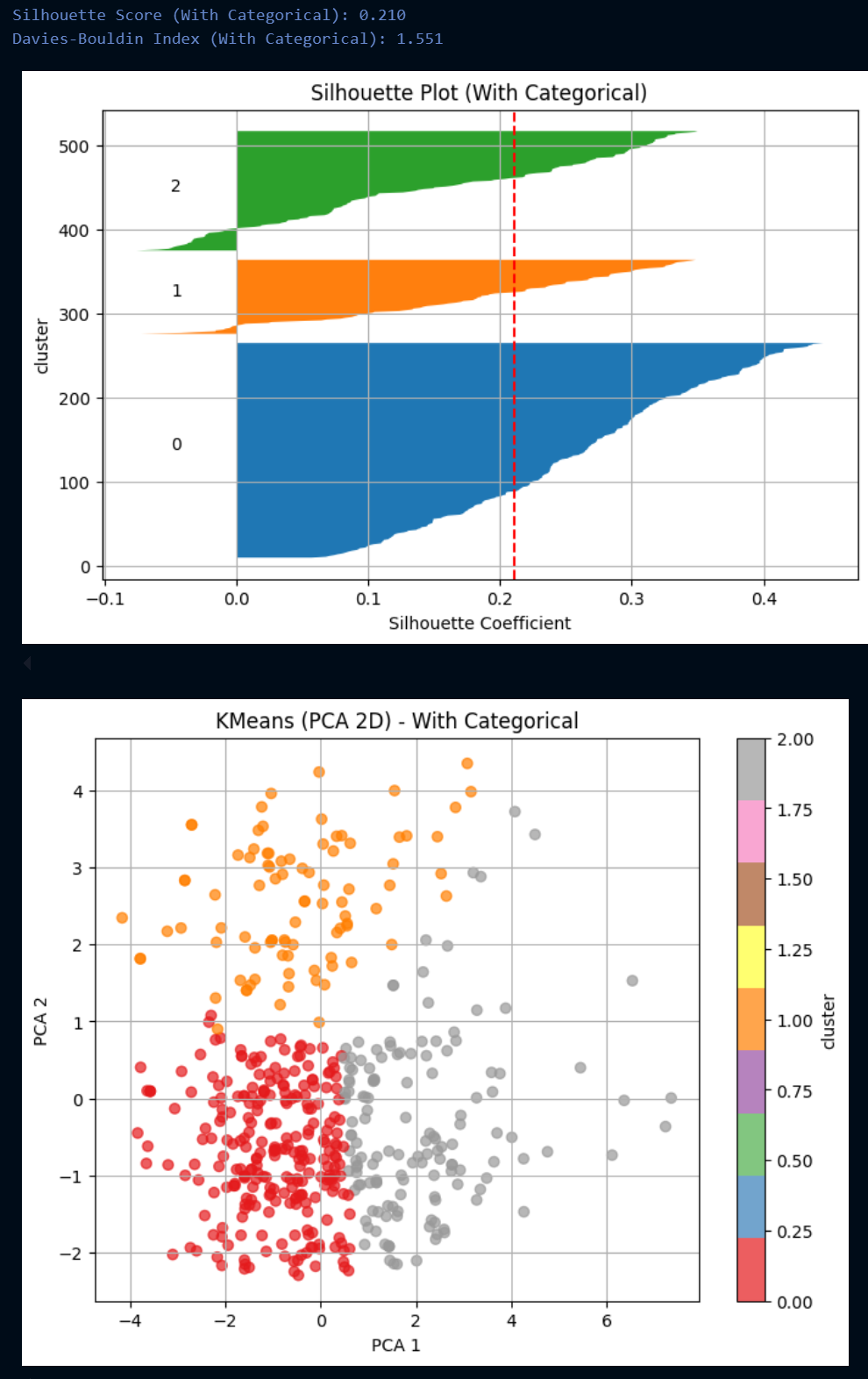
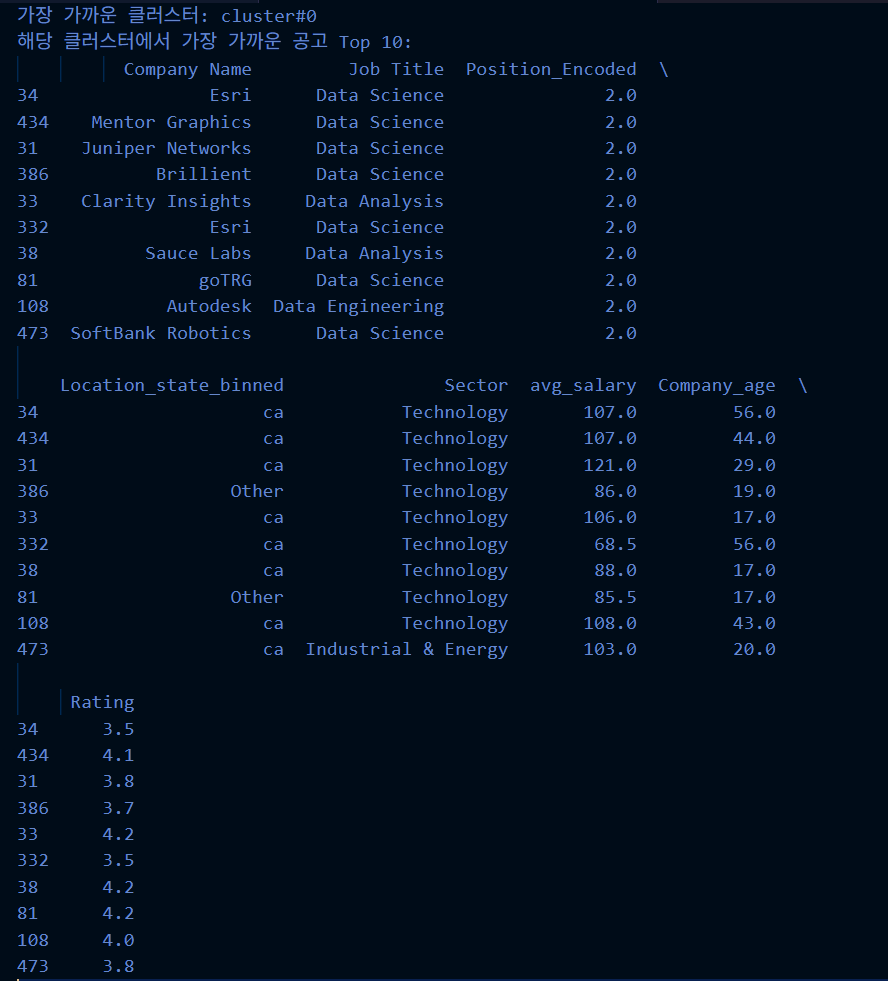
K-Means 클러스터링을 두 시나리오에 각각 적용했다. 클러스터 수(k)는 2~6 범위를 시도하고 Elbow Method로 급격한 기울기 완화 지점을 찾았다.

수치형만: Elbow가 k = 4 부근에서 꺾임.

범주 포함: Elbow가 k = 3 부근에서 완만해짐.

random\_state=42, n\_init=10, max\_iter=300을 공통 사용.

**Learning model evaluation and analysis**

**** ****    

클러스터 대표 샘플: 각 클러스터의 중심에서 가장 가까운 공고 Top 10을 추출해 ‘군집 프로파일’을 손쉽게 파악할 수 있도록 했다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **시나리오** | **최적 k** | **Silhouette** | **Davies-Bouldin** | **해석** |
| **수치형만** | **4** | **0.266** | **1.189** | **군집 간 분리가 상대적으로 뚜렷함** |
| **수치 + 범주** | **3** | **0.210** | **1.551** | **범주형 원핫 인코딩으로 거리 신뢰도가 떨어짐** |

Silhouette은 0.2 – 0.3 사이여서 완전한 분리라고 보긴 어렵지만, 수치형 전용 모델이 범주형 포함 모델보다 0.05p 이상 높았다.

Davies-Bouldin 지수도 수치형 전용이 낮아(=더 좋음) 범주형을 그대로 원핫 인코딩해 넣는 전략이 오히려 군집 품질을 약화시켰음을 보여 준다.

PCA 2-D 시각화에서는 수치형 모델이 4개의 구역으로 비교적 균등하게 퍼진 반면, 범주형 모델은 3개 구역이 명확히 비대칭 분포를 보였다.

결론: 현 세팅에서는 “수치형만 사용한 K-Means(k = 4)”가 가장 해석 가능하고 품질도 양호하다. 범주형까지 포함하려면 K-Prototypes 같은 혼합형 전용 알고리즘이 필요하다.

**Learning Experience**

**Difficulties encountered and how you have solved them**

**1. Feature Selection 혼란:**

모델링을 적용함에 있어서 feature selection의 어려움이 있었지만 다양한 피쳐들을 사용할 수 있도록 시나리오를 나누어 해결했습니다.

**2. 범주형 원핫 거리 한계:**

원핫인코딩 된 범주형 데이터에 유클리드 거리를 계산하면, 타겟과의 일치 여부는 확인 할 수 있지만, 항목 간 유사성을 반영할 수 없고, 차원의 저주 문제가 따라옵니다.

실제로 범주형에서 더 낮은 실루엣계수를 보여 거리계산 방식에서 범주형 피쳐의 한계를 재확인했고 추후 수치형과 범주형을 동시에 처리할 수 있는 K-prototype 클러스터링을 적용해 볼 계획입니다.

**3. 비지도 모델 평가지표 부족:**

비지도학습의 특성상 결과 평가가 어렵다는 한계가 있습니다. PCA를 통한 시각화로 결과를 정성평가, elbow method를 통해 최적 k를 탐색, shillouete score를 활용해 퀄리티를 측정했고 이에 더해 davies bouldin score를 탐구하고 적용하여 보완했습니다.

**What you have learned doing the project**

**1. ‘입력 없음’ 편향:**

사용자 입력용 더미값으로서 0을 넣으면, 편향이 발생합니다. '입력없음'의 개념을 평균값으로 처리할 수 있음을 배우고 적용했습니다.

**2. 전처리의 중요성:**

결측값의 처리나, 스케일링 등 전처리 방식을 변경할 때마다 결과가 크게 바뀌는 것을 확인했습니다. 이 과정을 통해 모델링보다도 전처리가 중요하다는 것을 실제로 확인했습니다.

**3. 시나리오 분리 실험을 통한 이점:**

시나리오 분리 실험이 feature 전략 비교에 효과적임을 체득했습니다.