스프린트 미션 #12

1. 서론

프로젝트 배경 및 목표

이번 미션에서는 자전거 대여 시스템 운영 데이터를 분석하여 대여 수요 예측 모델을 개발하고, 이를 통해 대여 시스템의 효율성을 높이고 사용자 만족도를 증가시키는 방법을 찾아보고자 한다.

자전거 공유 서비스는 시간대, 요일, 계절, 기상 조건 등 다양한 요인의 영향을 받으며 수요가 빠르게 변화한다. 따라서 이러한 복합적인 패턴을 정확하게 이해하고 예측하는 것은 자전거 배치 효율성 향상 및 사용자 만족도 증대에 매우 중요하다.

분석은 다음과 같은 단계로 접근하였다.

1. 데이터의 기본 특성과 변수 간 관계를 이해하고 주요 인사이트를 도출
2. 데이터 전처리 및 Feature Engineering을 통해 예측에 적합한 형태로 가공
3. 다양한 머신러닝 모델을 적용하고 RMSLE 기준으로 성능을 비교
4. 분석 과정 중 발견된 시행착오를 기록...

최종적으로 가장 적합한 수요 예측 모델을 선정하고자 한다.

2. 데이터 탐색 및 전처리

2.1 변수 유형 정리

분석에 활용한 데이터는 특정 도시에서 실제 운영 중인 자전거 대여 시스템의 기록으로, **train.csv** 및 **test.csv** 두 개의 파일로 구성되어 있다.

train.csv에는 자전거 대여 수량과 다양한 설명 변수들이 포함되어 있으며, **test.csv**에는 모델 예측 대상인 count 변수 없이 설명 변수만 포함되어 있다.

아래 데이터셋은 자전거 대여량(count)에 영향을 미칠 수 있는 시간, 기상, 휴일 여부 등 다양한 설명 변수를 포함하고 있다. 각 변수의 유형 및 기본 설명은 다음과 같다.

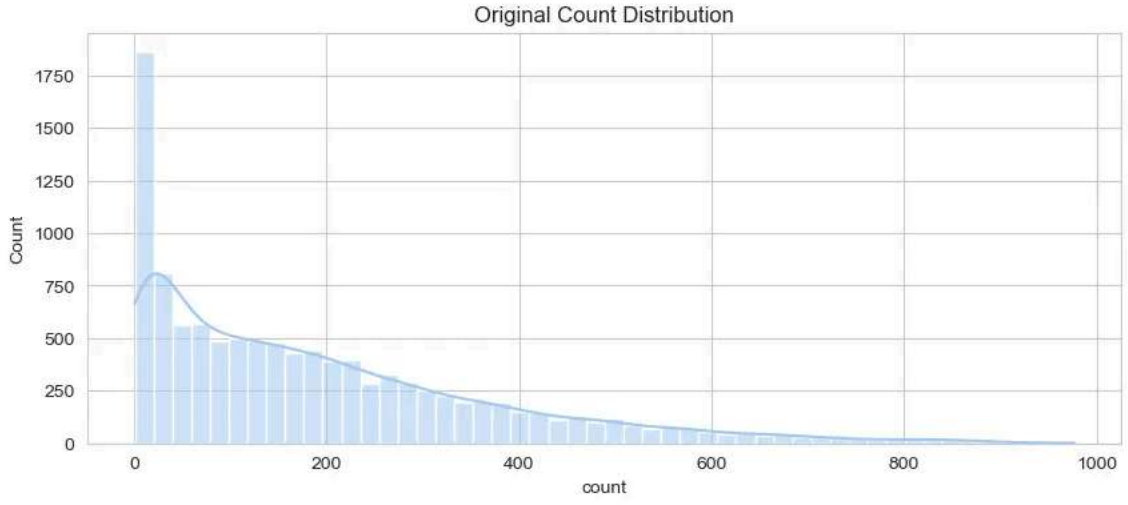
변수명	데이터 타입	설명	변수 유형
datetime	object	날짜 및 시간	시간 관련
season	int64	계절 (1: 봄, 2: 여름, 3: 가을, 4: 겨울)	범주형
holiday	int64	공휴일 여부 (0: 평일, 1: 공휴일)	범주형
workingday	int64	근무일 여부 (0: 주말/공휴일, 1: 근무일)	범주형
weather	int64	날씨 상태 (1: 맑음, 2: 구름낌/안개, 3: 약간의 비/눈, 4: 폭우/폭설)	범주형
temp	float64	실측 온도 (섭씨)	수치형
atemp	float64	체감 온도 (섭씨)	수치형
humidity	int64	습도 (%)	수치형
windspeed	float64	풍속 (m/s)	수치형
casual	int64	비회원 사용자 대여 수 (train.csv만 포함)	수치형
registered	int64	회원 사용자 대여 수 (train.csv만 포함)	수치형
count	int64	총 대여 수 (Target 변수)	수치형

2.2 결측치 및 0값 확인

데이터셋의 결측치를 먼저 확인한 결과, **train.csv와 test.csv 모두 결측치는 존재하지 않았다.**

```
python train.isnull().sum() test.isnull().sum() >>> datetime 0 season 0 holiday 0 workin
gday 0 weather 0 temp 0 atemp 0 humidity 0 windspeed 0 casual 0 registered 0 count 0 yea
r 0 month 0 day 0 hour 0 weekday 0 dtype: int64
```

또한, 각 변수에서 값이 **0**으로 기록된 경우가 많은 변수를 확인하였다. 특히 `windspeed` 변수에서 **0값이 빈번하게 나타났으며**, 이는 풍속의 실제 0이라기보다는 측정 오류 또는 기록 누락 가능성이 있다고 판단하였다.



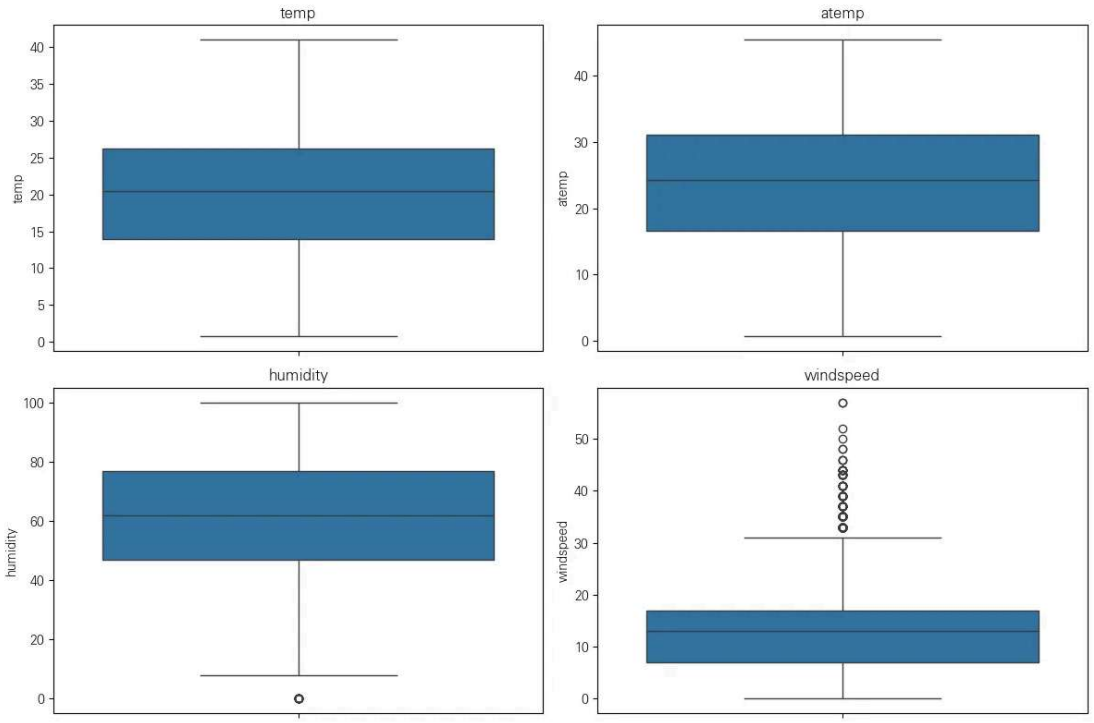
2.3 이상치 확인 및 처리

변수별 기초 통계량(describe) 확인 후, **boxplot 시각화**를 통해 이상치 존재 여부를 시각적으로 검토하였다.

이상치 확인 방법

```
python 복사편집 train.describe(include='all') sns.boxplot() # 변수별 boxplot
```

주요 이상치 확인 결과



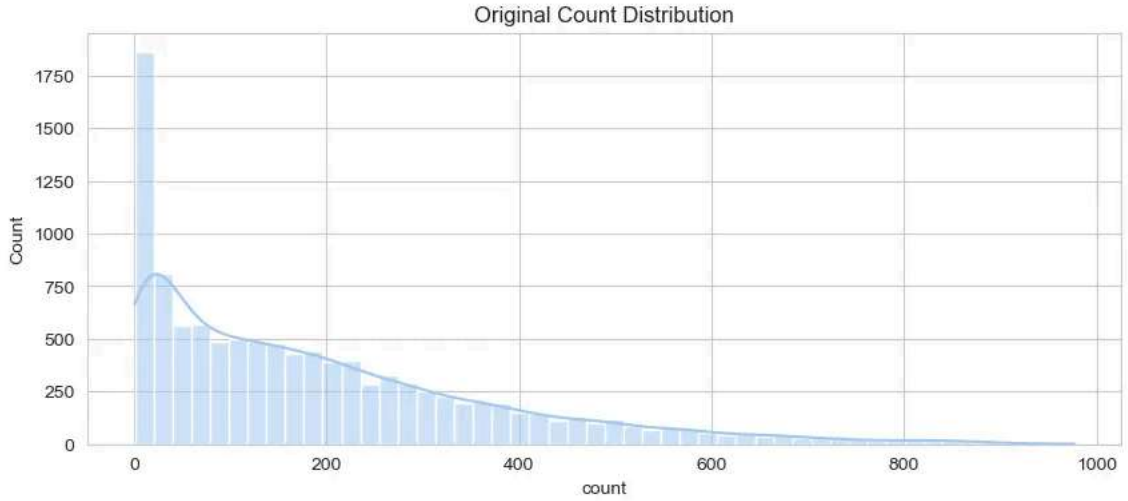
temp, atemp, humidity, windspeed

- temp와 atemp 변수는 전반적으로 **안정적인 분포**를 보였으며, 특이한 이상치는 발견되지 않았다.
- 반면, humidity와 windspeed 변수에서는 **극단적인 값이 확인되었다.**
  - humidity의 경우 일부 **매우 낮거나 높은 값**이 존재함. 절대 있을 수 없는 습도 0값도 존재하기에 이를 제외하고 구간화 시키기로 결정했다.
  - windspeed는 이상치로 보일만한 값이 여럿 있었으나 실제로 측정 가능한 범주이기 때문에 구간화 시키기로 결정했다.

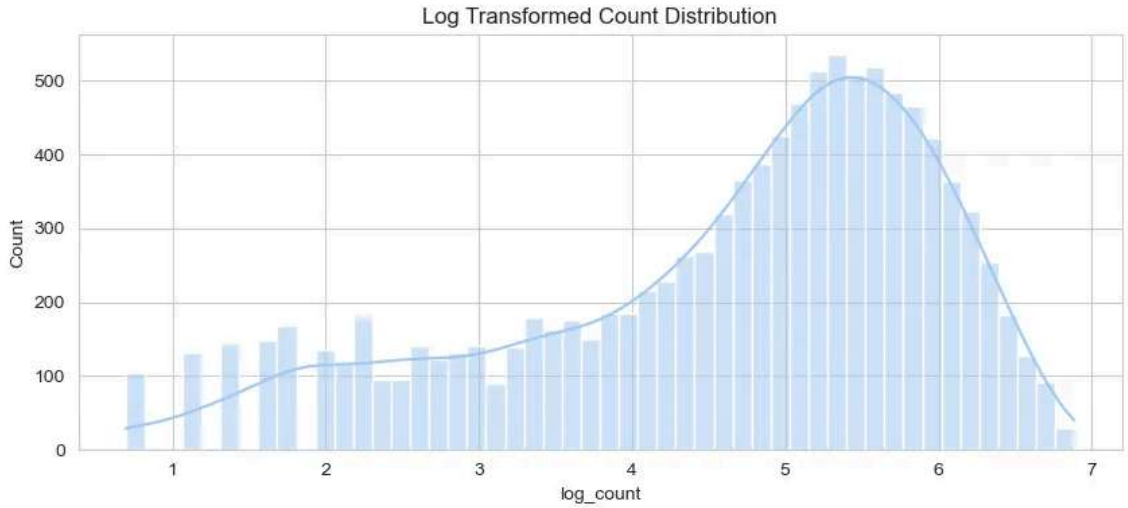
count (Target 변수)

- count 변수의 분포는 약 300~400 이상의 값부터 이상치로 판단할 수 있는 **고빈도 이상 값이 확인되었다**.
- 최대치는 약 1000까지 존재하였으며, 원본 분포가 **심한 오른쪽 왜도를 가지는 특성**을 보였다.
- 따라서 모델 학습의 안정성과 성능 향상을 위해 count 변수에는 **log(count + 1) 변환**을 적용하기로 결정하였다.

로그 변환 전 (오른쪽 왜도)



로그 변환 후 (살짝 치우친 정규 분포 모양을 보임)



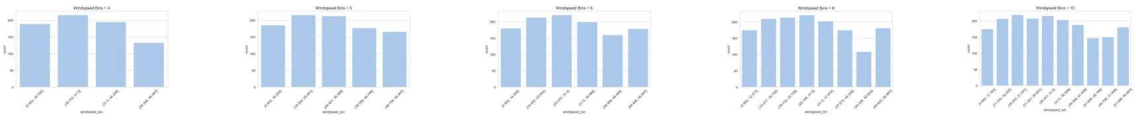
2.3.1 구간 매핑 처리

이상치 처리 및 변수 스케일 조정 목적으로, **humidity와 windspeed 변수에 대해 구간 매핑을 적용**하였다.

구간별 중앙값을 적용하여 모델 입력값으로 사용하였다.

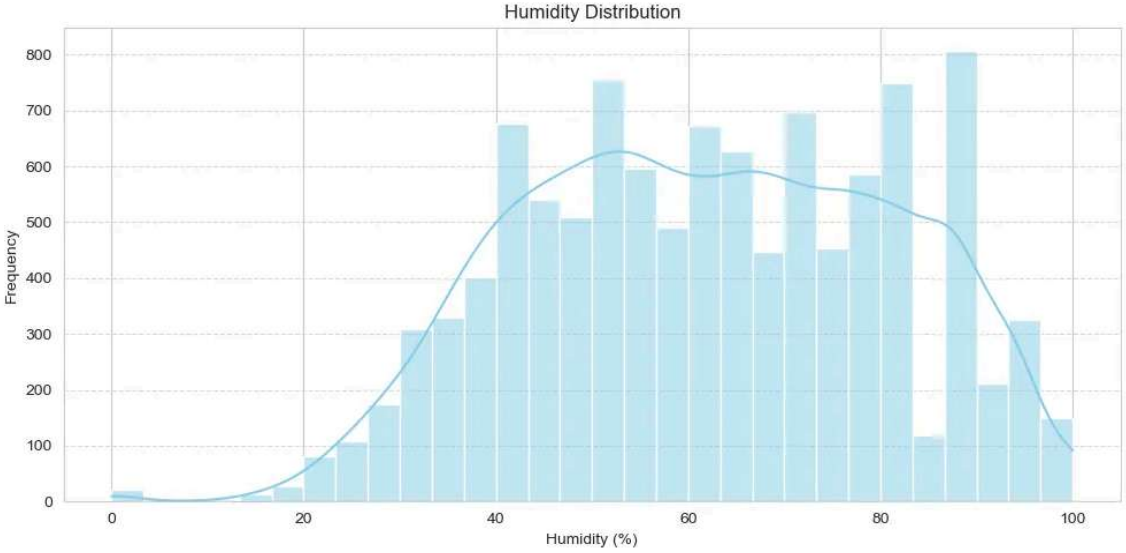
Windspeed

- 원본 windspeed 값은 **띄엄띄엄 존재**하는 특성을 보였으며, 특정 구간에 데이터가 밀집되어 있는 반면, 일부 극단적인 값은 매우 희소하게 존재하였다.
    - 이러한 분포 특성은 모델 학습 시 **해석 가능성 저하 및 일반화 성능 저하**로 이어질 수 있다고 판단되었다.
- 4, 5, 6, 8, 10 구간으로 시각화 분석을 수행한 결과, 6**구간 구간화가 가장 자연스러운 분포 반영** 및 성능 유지에 좋을 것이라 판단, 따라서 최종적으로 windspeed는 **6개 구간으로 구간화** 후 각 구간의 중앙값으로 변환하여 사용하였다.

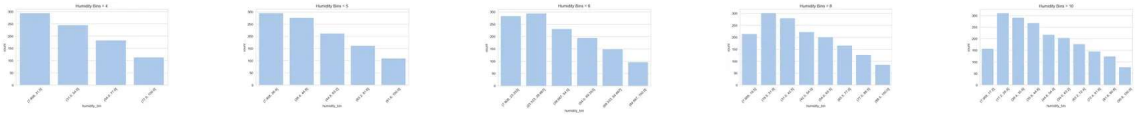


Humidity

- humidity 변수는 원본 값이 연속적이며, **20~60% 구간에 데이터가 집중**되어 있는 특성을 보였다.
- (추후 분석에서 다시 언급) 일부 극단적인 값은 존재하지만 IQR 기반 이상치 제거 적용 시 성능이 오히려 저하되는 결과가 나타났다. → **이상치 제거는 적용하지 않기로 결정**. 0값만 제거하여 사용하였다.



- 이후 모델 학습의 안정성을 확보하고 noise 영향을 완화하기 위해 **구간화를 적용하였다.**
- 4, 5, 6, 8, 10 구간으로 시각화 분석을 수행한 결과, **10구간 구간화가 가장 자연스러운 분포 반영** 및 성능 유지에 좋을 것이라 판단, 따라서 최종적으로 humidity는 **10개 구간으로 구간화** 후 각 구간의 중앙값으로 변환하여 사용하였다.

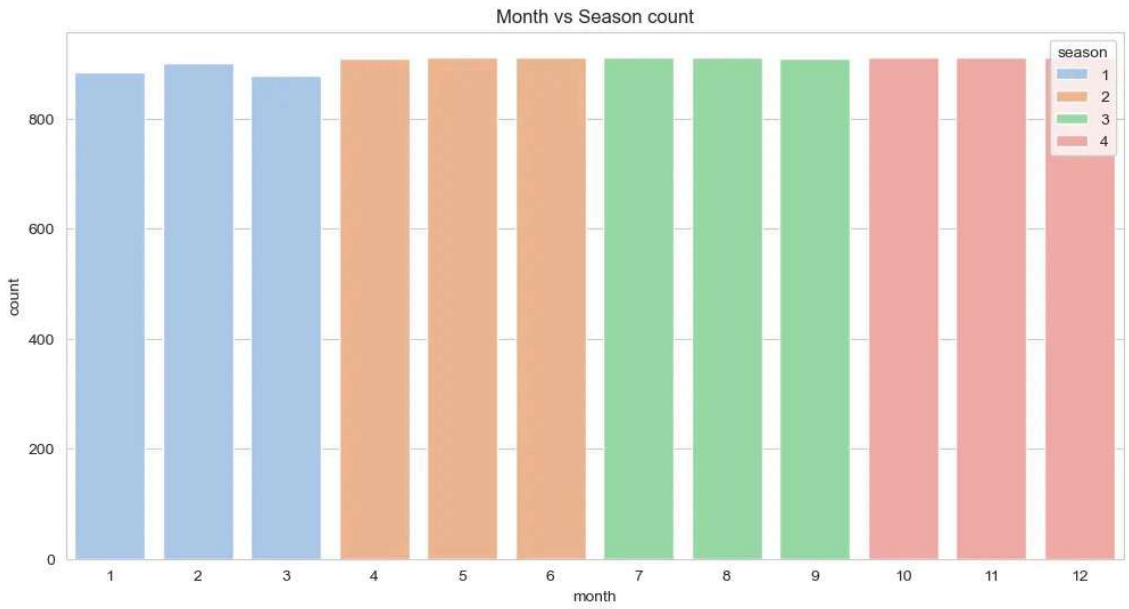


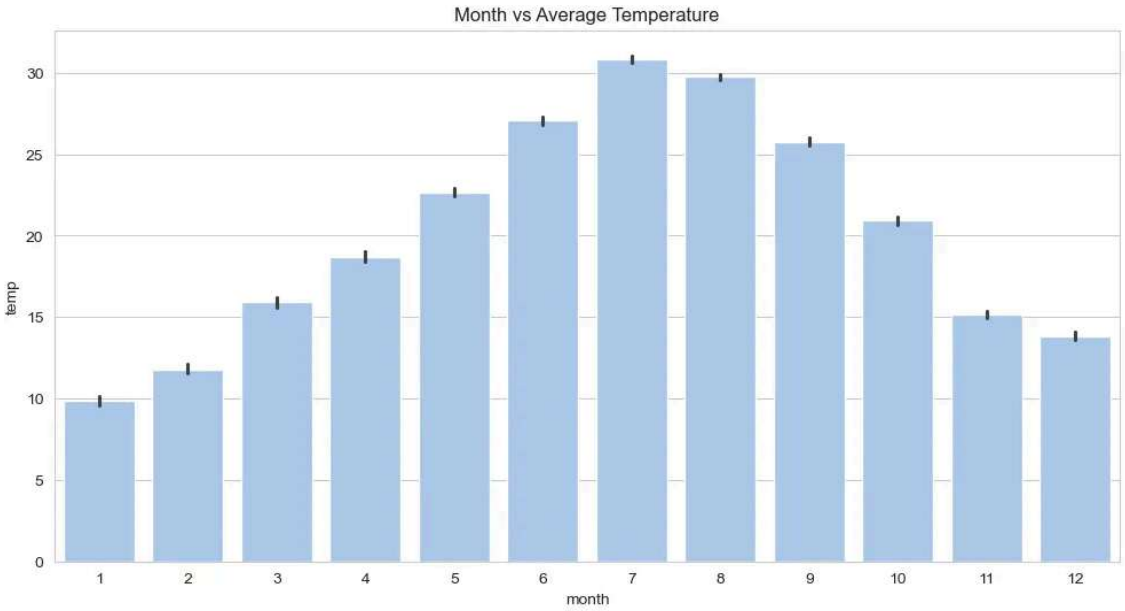
2.4 Season 컬럼

EDA 과정에서 season 컬럼 값이 실제 월(month)과 **일부 어긋나는 점**을 확인하였다.

시각화 분석 결과, 약 **2/3 정도의 레코드에서 일반적인 season 구분(봄: 3-5월, 여름: 6-8월, 가을: 9-11월, 겨울: 12-2월)과 다르게 season 값이 할당**되어 있음을 확인하였다.

예를 들어, 1월이 봄(season=1)으로 분류되는 등, **이를 바탕으로 계절 기반 데이터시각화는 어려움이 있을 것이라 본다. (ex, 봄에는 10%가 자전거를 탔다.)**





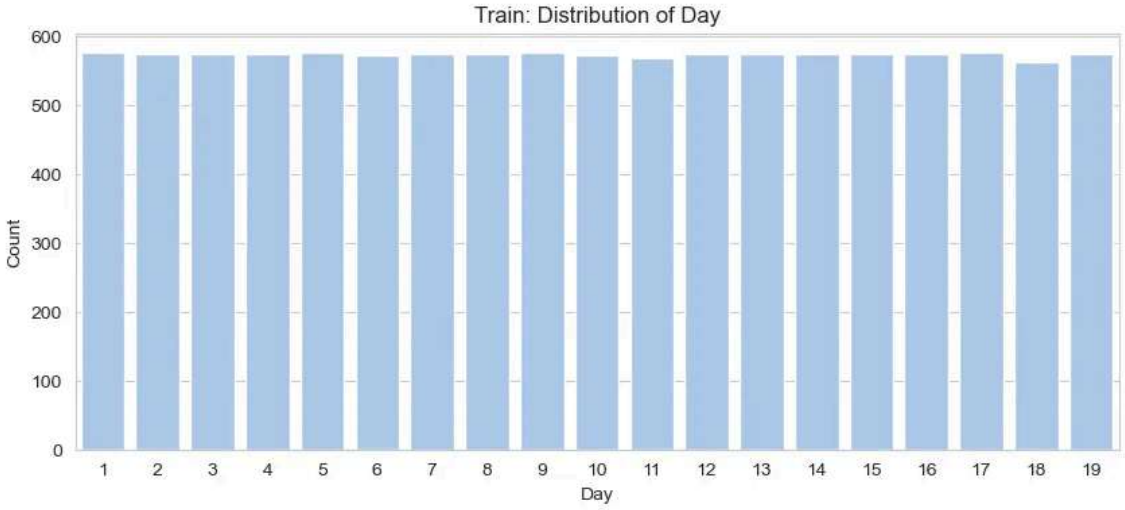
그럼에도 불구하고 **모델링 실험 결과, season 컬럼을 제거한 경우에 비해 포함한 경우 모델 성능이 개선**됨을 확인하였다. (이는 아래 분석에서 다시 언급)

2.5 Day Feature 분포 확인

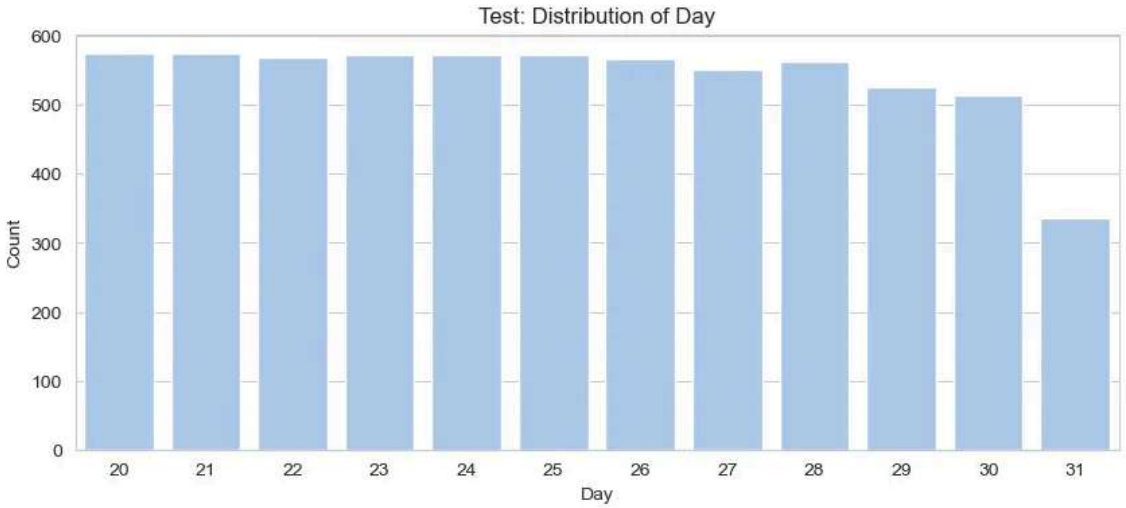
EDA 과정에서 day Feature의 분포를 확인한 결과, **train 데이터와 test 데이터 간에 분포 불일치가 존재함**을 확인하였다.

Train 데이터에서는 day 값이 **1일부터 19일까지 고르게 분포**하고 있었으나,

test 데이터에서는 day 값이 **20일부터 31일까지만 포함**되어 있으며, 일부 day 값에서는 샘플 수가 적게 나타나는 현상도 확인되었다.



`Train 데이터셋. 19일까지만 존재`



`Test 데이터셋. 20~31일 까지만 존재.`

이러한 분포 불일치는 모델 학습 시 train 데이터에서 학습한 day 관련 패턴이 test 데이터에 일반화되지 않을 위험이 있다.

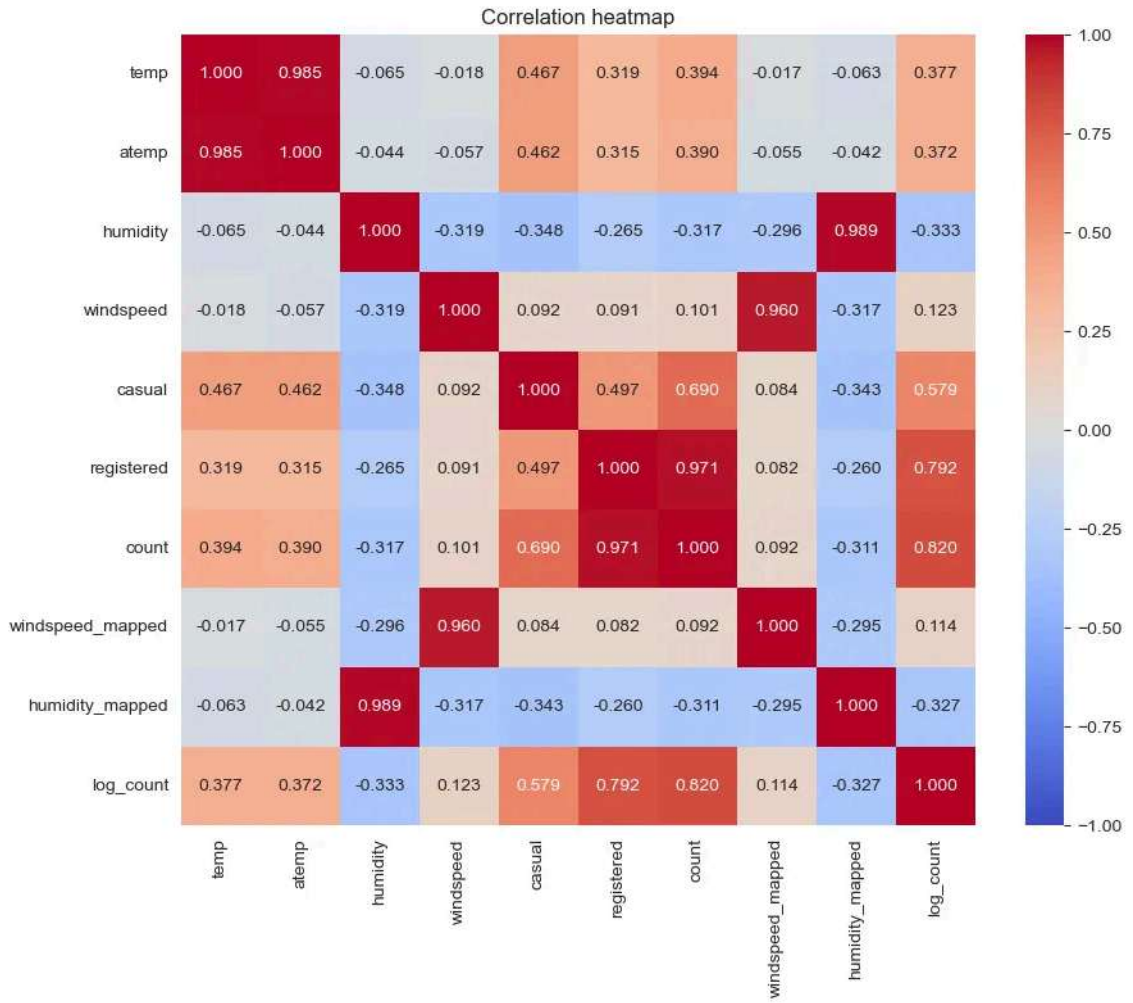
따라서 day Feature는 Feature selection 단계에서 **사용 여부를 신중히 검토**해야 할 Feature로 분류하였다. *(추후 분석에서 사용, 비사용 여부를 판단하였다.)*

2.6 상관관계 확인

EDA 과정에서 전체 변수 간 상관관계를 탐색하기 위해 **Pearson 상관계수 기반의 상관계수 히트맵을 시각화**하였다.

이 과정에서 temp(실측 온도)와 atemp(체감 온도) 변수 간 **상관계수가 과도하게 높은 수준(≈0.99 수준)을 보이는 점을 확인**하였다.

이는 두 변수가 거의 동일한 정보를 제공함을 의미하며, **모델 학습 시 다중공산성(multicollinearity) 문제를 유발할 가능성이 높다.**



이에 따라 temp와 atemp 변수 간의 **다중공산성 존재 여부를 정량적으로 확인**하기 위해 **Variance Inflation Factor (VIF) 분석**을 추가로 수행하였다.

Feature	VIF 값
const	9.57
temp	33.47
atemp	33.47

> 일반적으로 VIF 값이 10을 초과할 경우 다중공산성 문제로 간주하는데, 본 결과는 그 기준을 크게 초과하는 수준이었다.

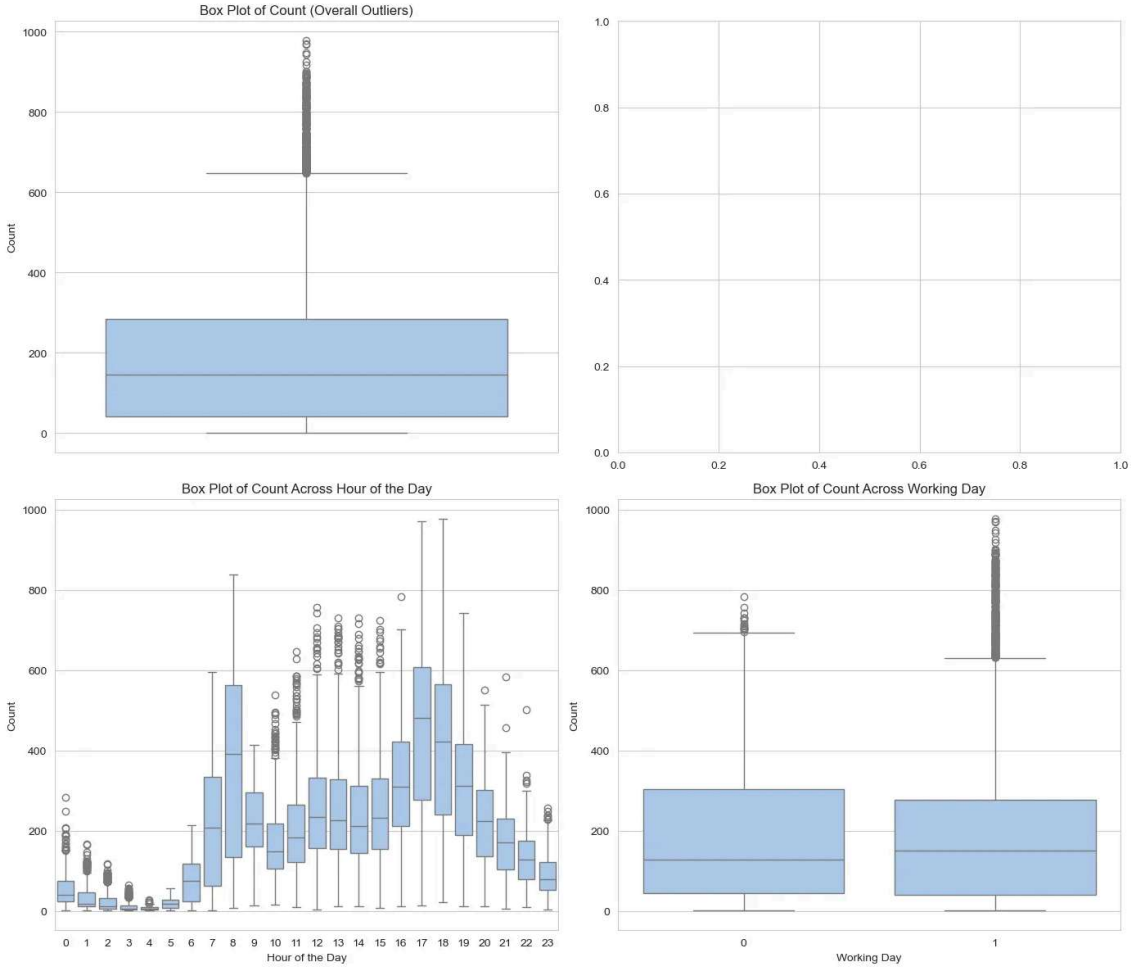
이후 모델링 실험 결과, **atemp 변수를 제거한 경우 모델 성능이 오히려 개선**됨이 확인되었으며, temp 변수만으로도 충분한 정보가 제공된다고 판단하였다.

따라서 최종 Feature 구성에서는 **temp 변수만 사용**하고, atemp 변수는 제거하기로 결정하였다.

2.7 Count 분포 및 주요 변수별 분석

EDA 과정에서 Target 변수인 count의 분포 및 주요 Feature 간 관계를 분석하였다.

Boxplot 및 분포 시각화를 통해 **이상치 존재 여부, 변수 간 주요 패턴, Feature 유효성**을 평가하였다.



전체 Count 이상치

- count 변수는 약 **300~400 이상부터 이상치로 간주할 수 있는 값이 확인되었다**.
- 최대치는 약 **1000까지 존재**하며, 전반적으로 **우측으로 긴 꼬리를 가지는 분포**를 보인다.
- 이러한 특성을 고려하여 **모델 안정성을 높이기 위해 log(count + 1) 변환**을 적용하기로 결정하였다.

Hour별 Count (가장 정보량이 많은 Feature)

- **출퇴근 시간대(8, 9시, 17, 18시)에 뚜렷한 수요 피크**가 확인되었다.
- **심야 및 새벽 시간대(0~5시)에는 수요가 매우 낮다**.
- hour Feature는 예측에 있어 상당히 중요하게 사용될 변수로 판단되었다.
- + 이런 식으로 일정 피크 부분에 추가 Featrue를 만들어 분석을 했더라면 결과가 조금이라도 더 나았지 않았을까 라는 아쉬움이 있다.

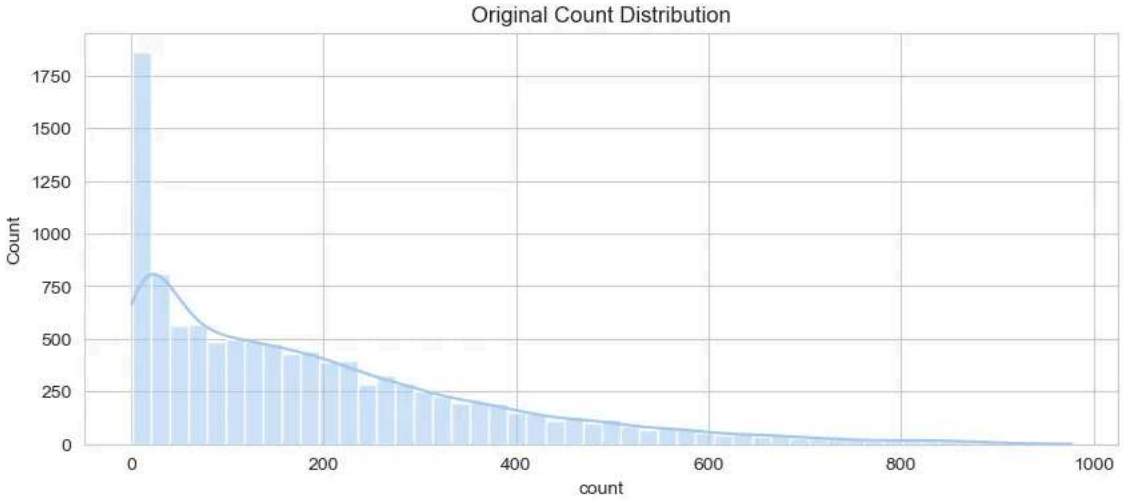
Workingday 여부

- 평일(workingday=1)에는 중앙값이 높고 사용량이 고르게 분포되어 있다. 높은 이상치(900대)도 다수 확인되었다.

- 휴일(workingday=0)에는 중앙값이 낮고 분산이 크다. **이용량의 양극화 경향**이 나타났다. (이용량이 높은 경우와 거의 없는 경우가 혼재)

2.8 Count 변수 로그 변환 적용

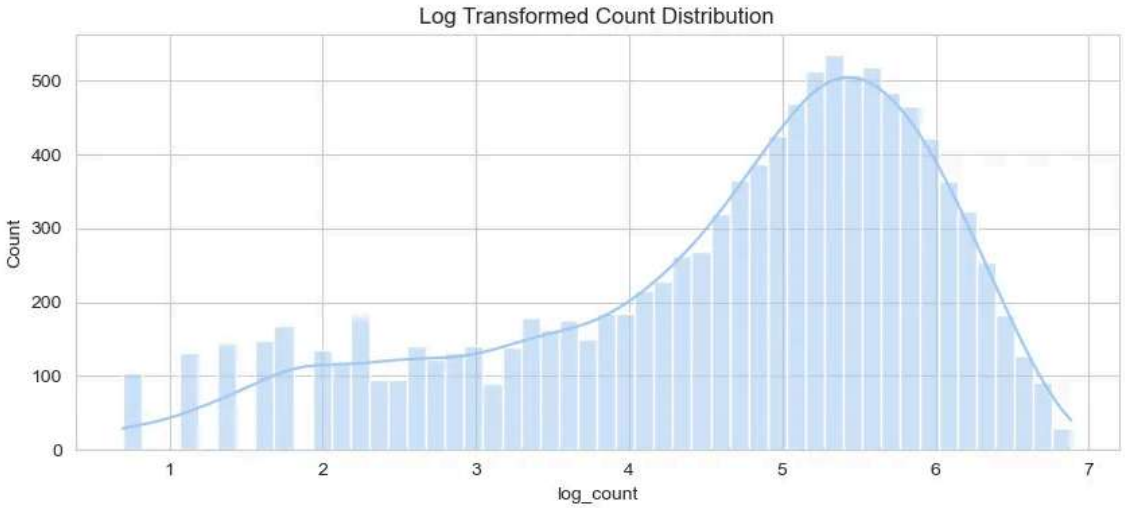
Target 변수인 자전거 대여량(count)은 EDA 결과 **오른쪽으로 긴 꼬리(Right-skewed)를 가지는 비정규 분포**임을 확인하였다.



이러한 분포 특성은 **높은 count 값에 모델이 과도하게 민감**하게 반응하게 만들며, 일반적인 패턴 학습 및 예측 성능에 부정적 영향을 미칠 수 있다.

따라서 count 변수에 대해 **log1p 변환**을 적용하여 분포를 보다 정규분포에 가깝게 변환하기로 결정하였다.

log1p 변환은 `log(count + 1)` 형태로 적용되며, **0 값을 안전하게 처리**할 수 있는 장점이 있다.



변환 후 count 변수의 분포는 **왜도(skewness)가 감소**했다.

> ☔ 다만, 로그 변환된 상태로 학습된 모델은 예측 시 **log1p 스케일의 값을 출력** 하게 되므로, 실제 count 단위로 해석하거나 제출하기 위해서는 반드시 **expm1 역변환**을 통해 원래 스케일로 복원하는 후처리 단계가 필요하다.

2.9 최종 Feature Engineering 결과

최종적으로 학습에 사용된 feature는 다음과 같다.

Feature명	처리 내용 / 변환 여부
datetime → year	연도 추출 (Feature 생성)
datetime → month	월 추출 (Feature 생성)
datetime → day	일 추출 (Feature 생성, 범위 불일치 있음)
datetime → hour	시 추출 (Feature 생성)
datetime → weekday	요일 추출 (Feature 생성)
season	월과 일부 불일치 확인 (원본 사용)
holiday	원본 사용
workingday	원본 사용
weather	원본 사용
temp	원본 사용
atemp	다중공산성 존재 → 제거
humidity	10구간 구간화 적용
windspeed	6구간 구간화 적용
casual	Target에 포함되는 정보 → 제거
registered	Target에 포함되는 정보 → 제거
count (Target)	**log1p 변환 적용 (학습 시), expm1 역변환 예정**

3. 모델링

본 단계에서는 앞선 EDA 과정을 통해 도출한 Feature Engineering 결과를 바탕으로, **자전거 대여량(count)의 수요 예측 모델을 개발**하고, 다양한 실험을 통해 최적의 모델 구성을 도출하는 것을 목표로 한다.

많은 모델을 돌려봤고, 상당히 많은 시행착오가 있었다. 때문에 최종 선정 모델과 이전의 시행착오 모델을 두 섹션으로나누어 작성하였다. (토글 참고)

▼ 3.1 모델링 시행착오

Baseline 모델 실험 결과

우선적으로 **Baseline 모델군(LinearRegression, Ridge, Lasso)** 를 통해 Feature 구성과 target 변수(log1p(count))의 관계를 평가하였다.

- Ridge RMSLE: **1.01444**
- Lasso RMSLE: **1.01445**

선형 모델 계열에서는 Feature들의 **비선형적 관계 특성**으로 인해 예측력이 매우 낮게 나타났으며, EDA 단계에서 확인한 Feature의 비선형적 특성상 선형 모델의 사용은 적합하지 않다고 판단하여 최종 모델 후보군에서 제외하였다.

RandomForest 실험 결과

비선형 관계를 잘 반영할 수 있는 RandomForest 모델은 **Baseline 모델 대비 매우 큰 성능 향상**을 보였다.

- RandomForest RMSLE (train): **0.10943**

시행착오

다만, **train set 을 나누지 않고 진행하여 과적합(overfitting) 이 일어났고,** 이를 검증하기 위해 교차 검증 기반 GridSearchCV 튜닝을 적용하였다. 최종적으로는 0.35대로 떨어졌다.

RandomForest 튜닝 결과

- 튜닝 전 RandomForest RMSLE: **0.10943 (train)** → 과적합
- 튜닝 후 RandomForest RMSLE (validation): **0.35170**

> 💡 결론:튜닝 전 값은 학습 데이터 기준 과적합 상태에서 나온 optimistic한 수치였으며, 튜닝 후 교차검증 기반 RMSLE가 실제 성능을 더 정확하게 반영한다고 판단되었다. 따라서 RandomForest는 최종 후보군에서는 제외하였다.

GradientBoosting 및 LightGBM 실험 결과

GradientBoosting 모델은 baseline 수준의 성능을 기록하였다.

- GradientBoosting RMSLE (validation): **0.3864**

XGBoost 실험 및 Hyperparameter Tuning

1차적으로 XGBoost 데이터셋도 마찬가지로 비율을 나누지 않고 진행하였는데, 0.1대의 낮은 값이 나왔다. 그 다음 분석에서는 8:2로 데이터셋을 나누어 별도의 튜닝 없이 모델링을 한 결과, 0.28268라는 우수한 결과가 나왔다.

이후 가장 유망한 몇 개의 모델을 위주로 추가 튜닝을 진행하기로 하였다. 아래는 그렇게 선별된 모델 중 XGBoost의 튜닝 결과다.

3.2.1 1차 튜닝 - XGBoost 튜닝 과정

GridSearchCV 기반 108개의 Hyperparameter 조합을 실험하였으며, 총 **324 fits**를 수행하였다.

- Best params:

  `{'colsample_bytree': 0.8, 'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 200, 'random_state': 42, 'subsample': 0.8}`
- Tuned XGBoost RMSLE (validation): **0.2770**

> 💡 튜닝 과정 주요 고려 사항
>
> learning_rate: 학습 안정성을 높이기 위해 낮은 값(0.1)으로 설정했다.
>
> max_depth: 과적합 방지를 위해 깊이를 5로 제한했다.
>
> subsample / colsample_bytree: 데이터 샘플링으로 모델 일반화 성능 확보를 시도했다.

3.2.2 2차 개선 튜닝 - Feature 수정 및 추가 파라미터 조정

1차 튜닝 결과를 바탕으로 추가적인 성능 개선 가능성을 검토하였다. EDA 분석과 Feature Engineering 결과를 반영하여 Feature list를 다음과 같이 재구성하고, 카테고리 Feature에 대해 **Label Encoding을 적용**하였다.

라벨 인코딩 대상 Feature

- humidity_mapped
- windspeed_mapped
- weekday

트리 기반 모델(XGBoost, LightGBM 등)은 본래 카테고리 Feature 처리가 가능하나, 실험의 일관성을 위해 동일하게 Label Encoding을 적용함.

- 최종 사용 Feature list:
  - holiday, workingday, weather, temp, season, humidity_mapped, windspeed_mapped, year, month, hour, weekday, day

또한 XGBoost 모델 구성 측면에서 **learning_rate를 더 낮추고 트리 수(n_estimators)를 대폭 증가**시키는 방식으로 **학습 안정성 및 성능 개선을 시도**하였다.

→ early_stopping 없이 3000 트리까지 학습 진행했다.

```
xgb.XGBRegressor( n_estimators=3000, learning_rate=0.03, max_depth=5,
min_child_weight=3, subsample=0.8, colsample_bytree=0.8, random_state=42,
tree_method='hist' )
```

학습 단계 (트리 수)	validation set RMSE (log scale)
0	1.40029
100	0.41744
500	0.27972
1000	0.26748
1500	0.26422
2000	0.26329
2500	0.26295
최종 (3000)	**0.26382**

최종 Validation RMSLE (expm1 역변환 적용 후 기준): **0.26382**

개선 포인트

- **learning_rate 감소(0.03)** → 보다 안정적인 학습 → **과적합 방지 및 성능 향상 효과** 확인
- **n_estimators 증가(3000)** → 충분한 학습 확보 → 최적 시점에서는 과적합 없이 꾸준한 개선 확인
- Feature Engineering 측면에서 **weekday / humidity_mapped / windspeed_mapped** 인코딩 적용이 **학습 안정성에 긍정적 기여**한 것으로 확인됨

> 💡 결론
> 1차 튜닝 대비 Validation RMSLE가 0.2770 → 0.2638로 개선전체 모델링 과정에서 가장 우수한 Validation 성능을 기록했다.
> Feature Engineering + 학습 파라미터 조정의 효과가 실제로 반영되었음을 확인했다.

최종 판단

- XGBoost 모델은 Feature Engineering 개선과 튜닝 전략 변경을 통해 성능을 안정적으로 개선 가능하였다.
- 최종적으로 XGBoost 튜닝 후 모델(2차 개선 버전)을 최종 단일 모델로 선정하여 **최종 제출 및 추가 실험의 기준 모델로 활용**하기로 결정하였다.

3.2.3 Feature Selection 추가 실험 결과

모델링 과정 후반부에서는 **Feature selection 과정에서 추가적인 실험을 통해 Feature 구성에 따른 성능 영향을 검토**하였다. 이는 앞서 진행했던 EDA와 Feature Engineering 결과와는 다소 상반되는 결과가 일부 나타났다.

실험 내용	Validation RMSLE 변화
day Feature 추가 전/후	0.269 → **0.265**
season Feature 추가 전/후	0.265 → **0.26382**
humidity IQR 이상치 제거 적용 시	**성능 저하** → 0.28333

- `day Feature`는 EDA 단계에서 **train/test 분포 불일치 문제**로 인해 Feature 사용 여부를 신중히 검토하였다. Train 데이터에서는 day 값이 **1~19일까지만 존재**하고, test 데이터에서는 **20~31일까지만 존재**하는 등 **명확한 분포 차이**가 확인되었기 때문이다.

  이는 일반적으로 모델이 day Feature에서 학습한 패턴이 test 데이터에 일반화되지 못할 위험성을 내포한다.

  > 그러나 실제 모델링 실험에서는 **day Feature를 추가 시 RMSLE가 개선**되는 결과가 나타났다.
  >
  > → 모델이 day Feature에서 **특정 시계열적 패턴이나 잔차적 패턴을 일부 학습**하여 예측 성능 향상에 기여한 것으로 판단된다.

  따라서 이러한 실험 결과를 바탕으로 day Feature는 **최종 Feature 구성에 포함**하였다.

- `season Feature`역시 EDA에서 월(month)과 불일치하는 문제가 관찰되었으나, **추가 시 성능이 더 향상**되는 결과를 보였다.

  → 이는 season Feature가 다른 Feature들과의 **비선형적 상호작용**을 통해 추가적인 예측 정보를 제공한 것으로 해석된다. (추측)

- 반면, `humidity` **변수에 대해 IQR 기반 이상치 제거 적용 시** 성능이 오히려 저하됨을 확인하였다.

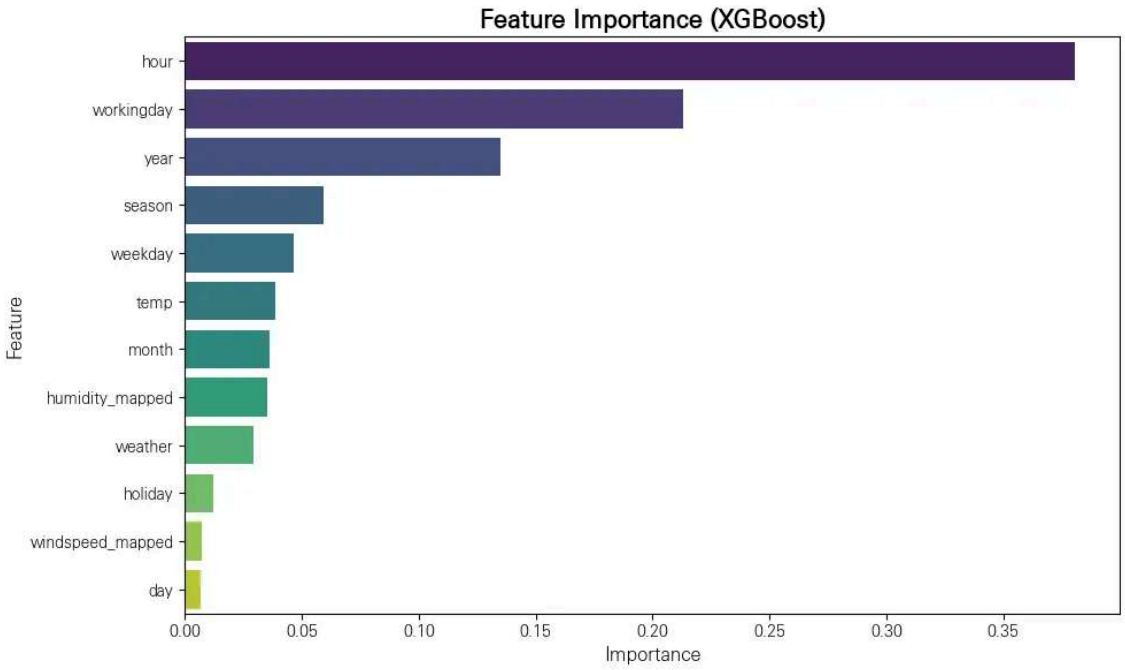
  → 이는 **extreme한 값들이 실제 예측에 유용한 정보로 작용**했을 가능성을 시사한다.

  따라서 최종 Feature 구성에서는 humidity 변수에 대해 **구간 매핑은 유지하되, 이상치 제거는 적용하지 않는 방향**으로 결정하였다.

3.3 Feature Importance 분석

최종 선정한 XGBoost 모델을 기준으로 **Feature Importance 분석을 수행**하였다.

아래는 모델 학습 결과 기준으로 계산된 Feature Importance 결과이다.



주요 결과 및 해석

- **hour** Feature가 압도적으로 높은 중요도를 기록하였다.

  → 이는 EDA 과정에서 확인한 **출퇴근 시간대 패턴**이 모델 예측에 매우 강한 영향을 미쳤기에 그랬으리라 추측할 수 있다.

- **workingday** Feature가 두 번째로 높은 중요도를 기록하였다.

  → 평일/휴일에 따른 자전거 수요 패턴 차이가 모델 학습에 중요한 역할을 하고 있음을 보여준다.

- **year** Feature의 중요도 역시 상당히 높게 나타났다.
- **season / weekday / temp / month / humidity_mapped** 등은 중간 수준의 중요도를 기록하였다.

  → 계절적 변화, 요일별 패턴, 온도 및 습도 구간화 Feature가 모델 성능에 기여하고 있음을 보여준다.

- **weather / holiday / windspeed_mapped / day** Feature들은 상대적으로 낮은 중요도를 기록하였다.

  향후 Feature selection이나 모델 경량화 시 **제거 가능성까지 고려**할 수 있는 후보가 아닐까.

3.4 모델 실험 요약

모델	RMSLE (평가 기준)	비고
Ridge Regression	1.01444 (train)	baseline 선형 모델
Lasso Regression	1.01445 (train)	baseline 선형 모델