NBA 2023-24 Analysis

> NBA 2023-24 시즌 데이터는 총 300개의 행과 31개의 열로 구성되어 있다. 데이터에는 각 팀의 시즌 기록과 관련된 다양한 통계 정보가 포함되어 있다. 주요 변수는 시즌(Season), 팀명(Team), 승리(W), 패리(L), 순위(Finish), 평균 연변(Age), 평균 희(Ht.), 평균 몸무게(Wt.) 등이며, 게임 수(G), 출전 시간(MP), 야투(FG, FGA, FG%), 3점슈터(3P, 3PA, 3P%), 2점슈터(2P, 2PA, 2P%), 자유투(FT, FTA, FT%), 공격 리반드(ORB), 수비 리반드(DRB), 총 리반드(TRB), 연결(ASSIST), 스틸(STL), 블록(BLK), 터널오버(TOV), 파운(PF), 드시(PTS) 등이 포함되어 있다. 이 데이터는 시즌별 팀 성과 분석, 선수들의 게임 기변 비교, 주요 게임 지표의 상관관계 분석 등을 수행하는 데 유용하다.

Loading Datasets

NBA 2023-24 시즌 데이터는 총 300개의 행과 31개의 열로 구성된 데이터셋으로, 각 팀의 시즌 기록과 관련된 다양한 통계 정보를 제공합니다. 주요 변수로는 시즌(Season), 팀명(Team), 승리(W), 패배(L), 순위(Finish), 평균 연령(Age), 평균 키(Ht.), 평균 몸무게(Wt.) 등이 있으며, 경기 수(G), 출전 시간(MP), 야투(FG, FGA, FG%), 3점슛(3P, 3PA, 3P%), 2점슛(2P, 2PA, 2P%), 자유투(FT, FTA, FT%), 공격 리바운드(ORB), 수비 리바운드(DRB), 총 리바운드(TRB), 어시스트(AST), 스틸(STL), 블록(BLK), 턴오버(TOV), 파울(PF), 득점(PTS) 등이 포함됩니다. 이 데이터는 시즌별 팀 성과 분석, 선수들의 경기력 비교, 주요 경기 지표의 상관관계 분석 등을 수행하는 데 유용합니다.

```
import pandas as pd
pd.set_option("display.max_columns", 8)

nba = pd.read_csv("NBA_2023-24.csv")
nba.info()
```

`NBA.head()`는 데이터프레임의 상위 5개 행을 미리보기 형태로 보여주는 함수로, 데이터의 구조와 내용을 빠르게 파악할 수 있도록 도와줍니다. 이를 통해 각 열(컬럼)의 데이터 유형, 예시 값, 범위 등을 확인할 수 있어 데이터 전처리 및 분석 계획을 수립하는 데 유용합니다. 예를 들어, NBA 2023-24 시즌 데이터의 경우, NBA.head()를 통해 팀명(Team), 시즌(Season), 승리(W), 패배(L) 등 주요 지표의 초기 데이터를 확인할 수 있으며, 데이터의 이상치나 결측치의 존재 여부를 빠르게 파악할 수 있습니다.

Data Cleaning and Preparation

`Ht.` 변수의 값은 `6'6"` 같은 문자열로 구성되어 있습니다. 이는 해석 및 회귀에 문제가 되면서 필요한 계산을 수행하기 힘들기 때문에 가능하면 해당 자료를 변환할 필요가 있습니다.

```
nba['Ht.'] = nba['Ht.'].apply(lambda x: int(x.split("'")[0]) * 12 + int
    (x.split("'")[1]))
```

몇가지 데이터의 형태를 확인해서 데이터 예측을 진행하도록 합니다.

```
numerical_columns = ['Age', 'Wt.', 'W', 'PTS', 'AST', 'REB']
nba[numerical_columns] = nba[numerical_columns].apply(pd.to_numeric,
    errors='coerce')
```

Linear Regression feature selection

> Feature Selection은 선형 회귀(Linear Regression) 모델의 성능을 향상시키기 위해 중요한 변수를 선택하고 불필요한 변수를 제거하는 과정입니다. 올바른 특성(Feature)을 선택하면 모델의 복잡도를 줄이고, 과적합(Overfitting)을 방지하며, 해석 가능한 모델을 만들 수 있습니다.

종속 변수(W, 승리 수)에 가장 큰 영향을 미치는 독립 변수(Feature) 5개를 선택하도록 하겠습니다.

- SelectKBest: 피처 선택을 위해 사용되는 함수로, 가장 중요한 K개의 피처를 선택합니다.
  - score_func=f_regression: f_regression 함수를 사용하여 각 독립 변수와 종속 변수(y) 간의 통계적 관계(선형 회귀)를 측정합니다.
  - k=5: 가장 중요한 5개의 피처를 선택하라는 의미입니다.
- f_regression은 피처의 F-값을 계산하고, p-값을 바탕으로 중요한 피처를 결정합니다.

```
X = nba.drop(["Season", "Team", "W", "Ht."], axis=1)
y = nba["W"]

from sklearn.feature_selection import SelectKBest, f_regression
features = SelectKBest(score_func=f_regression, k=5)
features.fit(X, y)
selectedFeatures = X.columns[features.get_support()]

X = nba[selectedFeatures]
y = nba["W"]
```

Performing the train, test, split-linear regression style

> 데이터를 훈련 세트와 테스트 세트로 분할합니다.

이 코드는 X, y 데이터를 훈련 세트(80%)와 테스트 세트(20%)로 분할합니다. 훈련 세트는 모델 학습에 사용되며, 테스트 세트는 모델 성능을 평가하는 데 사용됩니다. random_state=42는 데이터의 재현성을 보장하기 위해 사용됩니다. 이 코드는 머신러닝 모델을 학습하고 평가할 때 필수적인 절차입니다.

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size
    =0.2, random_state=42, stratify=y)
```

Making the model

LinearRegression()을 사용하여 선형 회귀 모델을 생성하고, fit() 메서드로 훈련 데이터(X_train, y_train)를 사용해 모델을 학습합니다. 각 피처의 회귀 계수를 확인하면, 특성이 종속 변수(W)와 어떻게 관계가 있는지 해석할 수 있습니다. 절편(Intercept)도 함께 출력되어, 독립 변수가 0일 때 종속 변수가 갖는 기본값을 알 수 있습니다. R^2스코어는 모델의 설명력을 나타내며, 1에 가까울수록 더 좋은 모델입니다. MSE (평균 제곱 오차)는 예측값과 실제값 간의 차이의 제곱합을 평균한 값으로, 작을수록 더 좋은 모델입니다.

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_percentage_error

nba_model = LinearRegression()
nba_model.fit(X_train, y_train)
y_predictions = nba_model.predict(X_test)
mean_absolute_percentage_error(y_test,y_predictions)
mean_absolute_percentage_error
```

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score,

rf_model = RandomForestRegressor(random_state=42)
rf_model.fit(X_train, y_train)
rf_predictions = rf_model.predict(X_test)
mse = mean_squared_error(y_test, y_predictions)
mae = mean_absolute_error(y_test, y_predictions)
r2 = r2_score(y_test, y_predictions)
print(f'MSE: {mse}, MAE: {mae}, R^2: {r2}')
```

Finding the coefficients and intercepts of the regression model equation

```
print(nba_model.coef_)
print(nba_model.intercept_)
```

Viewing results of predictions on a scatterplot

```
import matplotlib.pyplot as plt
import numpy as np

plt.scatter(y_test, y_predictions, color="red")
plt.plot(np.arange(min(y_test), max(y_test)), np.arange(min(y_test),
    max(y_test)), color="blue", linestyle="dashed")
plt.xlabel('Actual values')
plt.ylabel('Predicted values')
```

```
plt.title('Actual vs Predicted values')
plt.show()
```

중요도도 함께 확인해보도록 하겠습니다.

```
import matplotlib.pyplot as plt

importances = rf_model.feature_importances_
feature_importance_df = pd.DataFrame({'Feature': X_train.columns, '
    Importance': importances})
feature_importance_df = feature_importance_df.sort_values(by='
    Importance', ascending=False)

plt.barh(feature_importance_df['Feature'], feature_importance_df['
    Importance'])
plt.xlabel('Feature Importance')
plt.title('Feature Importance for Random Forest')
plt.show()
```