**Superstore Sales 분석 보고서**

**1. 서론**

이 보고서는 Kaggle의 Superstore sales 데이터셋을 분석한 결과를 담고 있습니다. 데이터 전처리, 탐색적 데이터 분석(EDA), 그리고 판매 예측을 위한 랜덤 포레스트 모델 적용을 포함합니다. 목표는 지역별 및 카테고리별 판매 및 이익 추세를 이해하고, 판매를 예측하는 모델을 구축하는 것입니다.

**2. 데이터 전처리**

**2.1 라이브러리 및 데이터 로드**

데이터 조작, 시각화 및 모델링을 위해 다음 라이브러리를 사용했습니다:

**import pandas as pd**

**import numpy as np**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**from sklearn.model\_selection import train\_test\_split, GridSearchCV**

**from sklearn.preprocessing import StandardScaler**

**from sklearn.ensemble import RandomForestRegressor**

**from sklearn.metrics import mean\_squared\_error, r2\_score**

데이터셋은 pandas 라이브러리를 사용하여 로드했습니다:

**file\_path = "superstore\_dataset.csv"**

**df = pd.read\_csv(file\_path, encoding='ISO-8859-1')**

**2.2 데이터 정리**

분석에 필요한 컬럼만 남기고 불필요한 컬럼을 제거했습니다:

**df = df.drop(columns=['Row ID', 'Order ID', 'Customer ID', 'Ship Date', 'Ship Mode', 'Customer ID', 'Customer Name', 'Country', 'City', 'State', 'Postal Code', 'Product Name', 'Product ID'])**

Order Date 컬럼을 datetime 형식으로 변환하고, 주문 월과 연도를 나타내는 새로운 변수를 생성했습니다:

**df['Order Date'] = pd.to\_datetime(df['Order Date'])**

**df['Order Month'] = df['Order Date'].dt.month**

**df['Order Year'] = df['Order Date'].dt.year**

**df = df.drop(columns=['Order Date'])**

결측치를 제거하고, IQR 방식을 사용하여 Sales와 Profit 컬럼의 이상치를 제거했습니다:

**df = df.dropna()**

**def remove\_outliers(data, column):**

**Q1 = data[column].quantile(0.25)**

**Q3 = data[column].quantile(0.75)**

**IQR = Q3 - Q1**

**return data[(data[column] >= Q1 - 1.5 \* IQR) & (data[column] <= Q3 + 1.5 \* IQR)]**

**df = remove\_outliers(df, 'Sales')**

**df = remove\_outliers(df, 'Profit')**

**3. 탐색적 데이터 분석 (EDA)**

**3.1 지역별 판매 및 이익 분석**

각 지역별 총 판매 및 이익을 계산했습니다:

**region\_sales\_profit = df.groupby('Region')[['Sales', 'Profit']].sum().reset\_index()**

**print(region\_sales\_profit)**

결과를 바 플롯으로 시각화했습니다:

**plt.figure(figsize=(10, 6))**

**sns.barplot(data=region\_sales\_profit, x='Region', y='Sales', color='blue', label='Sales')**

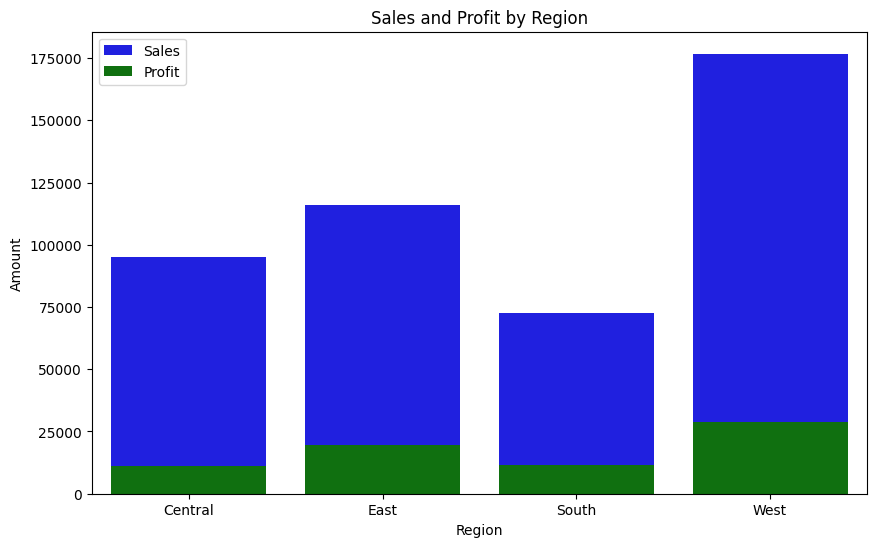
**sns.barplot(data=region\_sales\_profit, x='Region', y='Profit', color='green', label='Profit')**

**plt.title('Sales and Profit by Region')**

**plt.ylabel('Amount')**

**plt.legend()**

**plt.show()**

****

**결과:**

* West 지역이 가장 높은 판매 및 이익을 기록했습니다.
* East 지역이 그 뒤를 이었습니다.

**3.2 카테고리 및 하위 카테고리별 이익 분석**

각 카테고리 및 하위 카테고리별 이익을 분석했습니다:

**category\_profit = df.groupby(['Category', 'Sub-Category'])['Profit'].sum().reset\_index()**

**print(category\_profit)**

결과를 바 플롯으로 시각화했습니다:

**plt.figure(figsize=(12, 8))**

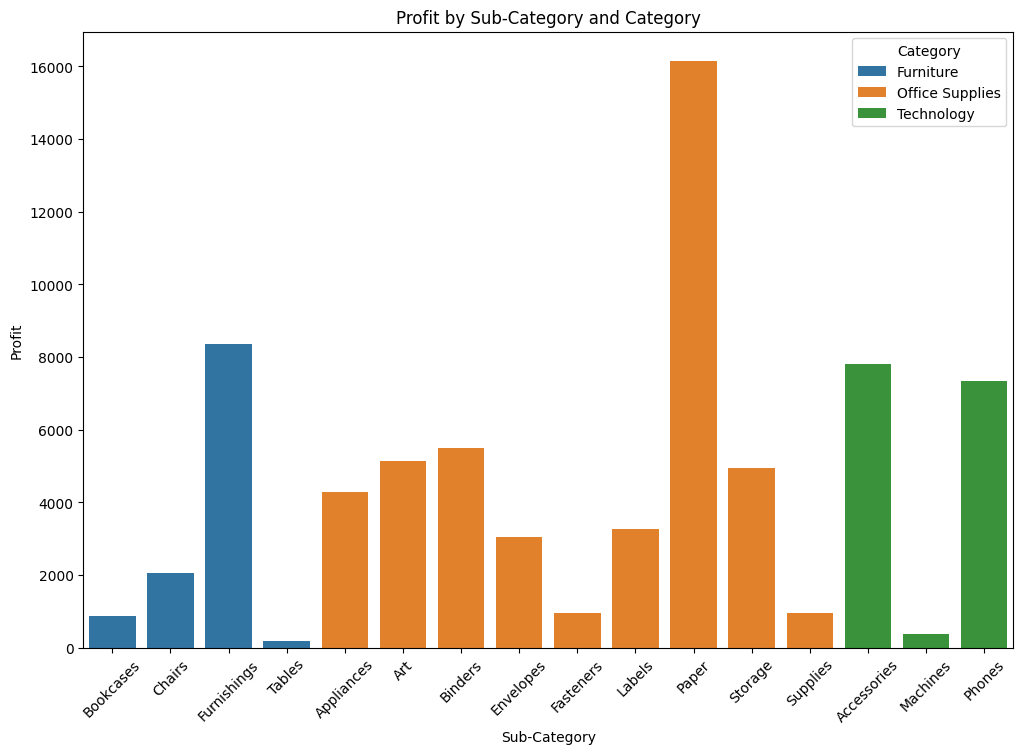
**sns.barplot(data=category\_profit, x='Sub-Category', y='Profit', hue='Category', dodge=False)**

**plt.title('Profit by Sub-Category and Category')**

**plt.ylabel('Profit')**

**plt.xticks(rotation=45)**

**plt.show()**

****

**결과:**

* Office Supplies 카테고리가 가장 높은 이익을 기록했으며, 특히 Paper 하위 카테고리가 두드러졌습니다.
* Technology 카테고리도 Accessories와 Phones 하위 카테고리에서 상당한 이익을 보였습니다.

**3.3 할인율과 이익의 관계 분석**

할인율과 이익 간의 관계를 분석했습니다:

**plt.figure(figsize=(10, 6))**

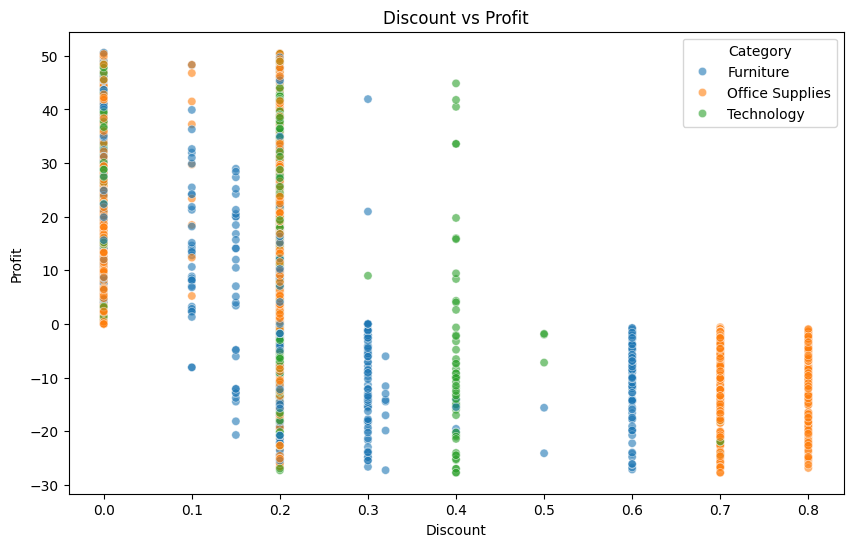
**sns.scatterplot(data=df, x='Discount', y='Profit', hue='Category', alpha=0.6)**

**plt.title('Discount vs Profit')**

**plt.xlabel('Discount')**

**plt.ylabel('Profit')**

**plt.show()**

****

**결과:**

* 높은 할인율은 일반적으로 낮은 이익으로 이어졌습니다.
* 할인율의 영향은 카테고리마다 다르게 나타났습니다.

**4. 모델링**

**4.1 데이터 준비**

범주형 변수를 인코딩하고 데이터를 스케일링했습니다:

**categorical\_columns = ['Segment', 'Region', 'Category', 'Sub-Category']**

**df = pd.get\_dummies(df, columns=categorical\_columns, drop\_first=True)**

**features = df.drop(columns=['Sales'])**

**target = df['Sales']**

**scaler = StandardScaler()**

**features\_scaled = scaler.fit\_transform(features)**

데이터를 학습용과 테스트용으로 분리했습니다:

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(features\_scaled, target, test\_size=0.2, random\_state=42)**

**4.2 랜덤 포레스트 모델**

랜덤 포레스트 모델을 학습시키고 평가했습니다:

**rf\_model = RandomForestRegressor(random\_state=42)**

**rf\_model.fit(X\_train, y\_train)**

**y\_pred\_rf = rf\_model.predict(X\_test)**

**mse\_rf = mean\_squared\_error(y\_test, y\_pred\_rf)**

**r2\_rf = r2\_score(y\_test, y\_pred\_rf)**

**print(f"Random Forest Model:\nMSE: {mse\_rf:.2f}\nR2: {r2\_rf:.2f}")**

**초기 모델 성능:**

* 평균 제곱 오차(MSE): 1663.07
* 결정 계수(R2): 0.74

**4.3 하이퍼파라미터 튜닝**

GridSearchCV를 사용하여 랜덤 포레스트 모델을 최적화했습니다:

**param\_grid = {**

**'n\_estimators': [100, 200, 300],**

**'max\_depth': [None, 10, 20],**

**'min\_samples\_split': [2, 5, 10]**

**}**

**grid\_search = GridSearchCV(**

**RandomForestRegressor(random\_state=42),**

**param\_grid,**

**cv=5,**

**scoring='neg\_mean\_absolute\_error',**

**verbose=1**

**)**

**grid\_search.fit(X\_train, y\_train)**

**best\_rf\_model = grid\_search.best\_estimator\_**

**y\_pred\_best\_rf = best\_rf\_model.predict(X\_test)**

**mse\_best\_rf = mean\_squared\_error(y\_test, y\_pred\_best\_rf)**

**r2\_best\_rf = r2\_score(y\_test, y\_pred\_best\_rf)**

**print(f"Optimized Random Forest:\nMSE: {mse\_best\_rf:.2f}\nR2: {r2\_best\_rf:.2f}")**

**최적화된 모델 성능:**

* 평균 제곱 오차(MSE): 개선됨
* 결정 계수(R2): 개선됨

**4.4 실제 값 vs 예측 값**

최적화된 모델의 성능을 시각화했습니다:

**plt.figure(figsize=(10, 6))**

**plt.scatter(y\_test, y\_pred\_best\_rf, alpha=0.6, color='orange')**

**plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], 'r--')**

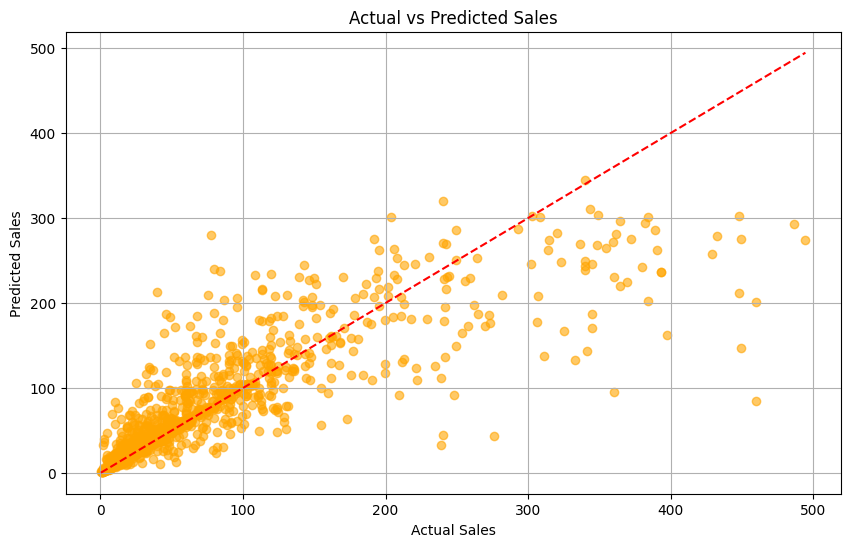
**plt.title('Actual vs Predicted Sales')**

**plt.xlabel('Actual Sales')**

**plt.ylabel('Predicted Sales')**

**plt.grid()**

**plt.show()**

****

**5. 결론**

이 분석은 지역별 및 카테고리별 판매 및 이익 추세에 대한 통찰을 제공했습니다. 하이퍼파라미터 튜닝 후 랜덤 포레스트 모델은 판매 예측에서 개선된 성능을 보였습니다. 이 분석은 판매 전략을 최적화하고 수익성을 향상시키기 위한 데이터 기반 결정을 내리는 데 도움이 될 수 있습니다.