TUGAS 2 - MACHINE LEARNING
Linear dan Polynomial Regression

KELOMPOK 9

- Fazhira Rizky Harmayani (2208107010012)
- Cut Dahliana (2208107010027)
- Naufal Aqil (2208107010043)
- Hidayat Nur Hakim (2208107010063)
- Riska Haqika Situmorang (2208107010086)

Pemahaman Dataset

Nama Dataset :

E-commerce Sales Prediction Dataset

Sumber Kaggle :

https://www.kaggle.com/datasets/nevildhinoja/e-commerce-sales-prediction-dataset

Deskripsi Dataset :

Dataset ini merupakan kumpulan data dirancang untuk analisis tren, strategi harga, dan prediksi penjualan. Dataset ini berisi informasi tentang produk yang dijual secara online, mencakup harga, rating, jumlah ulasan, diskon, dan lainnya.

Jumlah Data :

Dataset ini terdiri dari 1000 baris dan 7 kolom yaitu beberapa variabel yang merepresentasikan informasi produk e-commerce. ProductID (ID unik untuk setiap produk), sementara ProductName (nama produk). Category (kategori produk), Price (harga produk), dan Rating (penilaian pelanggan terhadap produk). Jumlah ulasan tercermin dalam NumReviews (jumlah ulasan), sedangkan StockQuantity (jumlah stok yang tersedia). Discount (potongan harga dalam bentuk persentase). Sales (jumlah produk yang terjual) merupakan target prediksi. DateAdded (tanggal produk ditambahkan ke dalam sistem), dan City (kota asal produk).

Variabel Dependen (Target) : Sales

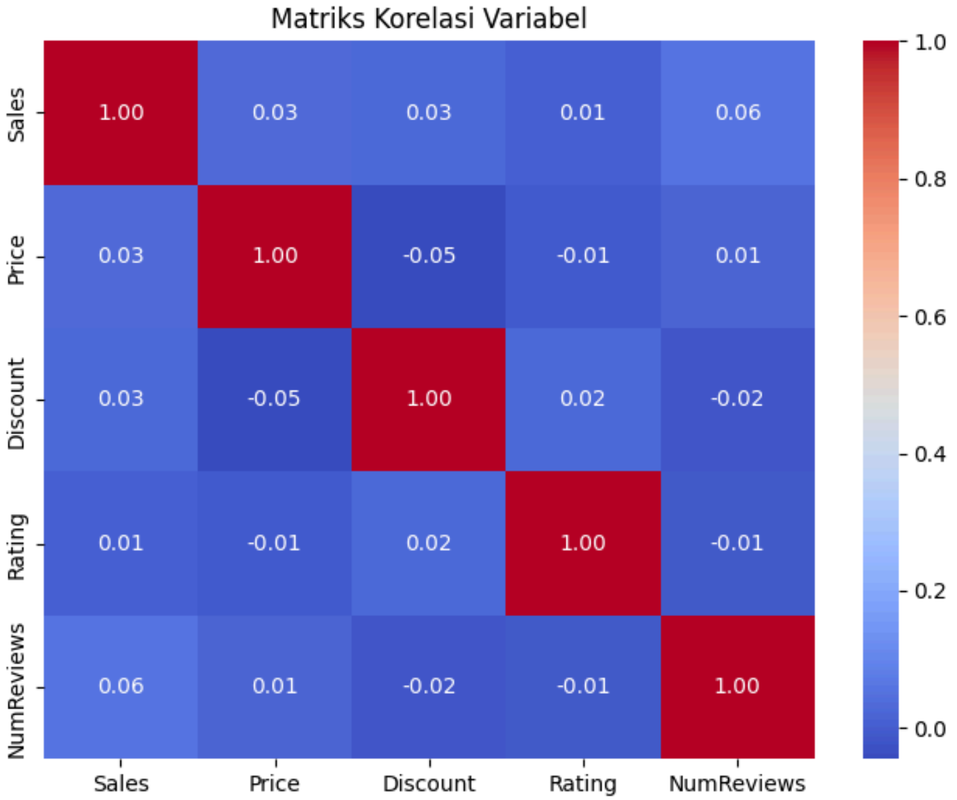
Variabel Independen (Fitur): Price, Rating, NumReviews, dan Discount

📈 Statistik Deskriptif

	ProductID	Price	Rating	NumReviews	StockQuantity	\
count	1000.000000	1000.00000	1000.000000	1000.000000	1000.000000	
mean	500.500000	253.77551	3.025600	2498.753000	495.395000	
std	288.819436	141.40362	1.151004	1463.241871	292.799253	
min	1.000000	10.11000	1.000000	3.000000	0.000000	
25%	250.750000	133.09250	2.100000	1201.750000	241.750000	
50%	500.500000	251.31000	3.100000	2476.000000	505.000000	
75%	750.250000	375.82750	4.000000	3797.500000	743.500000	
max	1000.000000	499.74000	5.000000	4994.000000	993.000000	

	Discount	Sales
count	1000.000000	1000.000000
mean	0.251640	1011.037000
std	0.146455	582.113466
min	0.000000	0.000000
25%	0.130000	502.000000
50%	0.250000	998.000000
75%	0.380000	1540.000000
max	0.500000	1997.000000

📊 Heatmap Korelasi



Eksplorasi Data dan Pra-pemrosesan

- **Memilih Kolom yang Relevan untuk Analisis**

```
# Pilih hanya kolom yang relevan untuk analisis
selected_columns = ["Sales", "Price", "Discount", "Rating", "NumReviews"]
df_clean = df[selected_columns].copy()
```

Kita hanya menggunakan beberapa kolom yang dianggap mempengaruhi Sales (jumlah produk terjual), yaitu Price (harga produk), Discount (diskon yang diberikan), Rating (penilaian produk), dan NumReviews (jumlah ulasan). Kolom-kolom ini dipilih karena memiliki potensi besar dalam memengaruhi keputusan pembelian konsumen.

- **Standarisasi Fitur Numerik**

```
scaler = StandardScaler()
numeric_features = ["Price", "Discount", "Rating", "NumReviews"]
df_clean[numeric_features] = scaler.fit_transform(df_clean[numeric_features])
```

Menggunakan StandardScaler() untuk menstandarisasi fitur numerik

- **Mengecek mising value**

```
Missing values per kolom:
 Sales         0
Price          0
Discount       0
Rating         0
NumReviews     0
dtype: int64
```

tidak terdapat missing value pada dataset ini

- **Mengecek outlier**

```
Jumlah outlier per kolom:
 Sales         0
Price          0
Discount       0
Rating         0
NumReviews     0
dtype: int64
```

tidak terdapat outlier pada dataset ini

Implementasi Model

kita akan membangun model prediksi Sales menggunakan dua pendekatan:

1. Linear Regression
2. Polynomial Regression

- **Pisahkan Fitur (X) dan Target (y)**

```
# Pisahkan fitur (X) dan target (y)
X = df_clean.drop(columns=["Sales"])  # Fitur: Price, Discount, Rating, NumReviews, Price_Discount
y = df_clean["Sales"]  # Target: Sales
```

- **Bagi Data menjadi Training & Testing (80%-20%)**

```
# Bagi data menjadi training & testing (80% - 20%)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

- **Inisialisasi dan Pelatihan Model Linear Regression**

```
#  Inisialisasi model regresi linear
linear_model = LinearRegression()

# 🔧 Latih model dengan data training
linear_model.fit(X_train, y_train)
```

Model Linear Regression dibangun untuk mempelajari hubungan antara fitur dan target.

- **Koefisien Model**

```
Koefisien Model Linear Regression:
                Koefisien
Price          -28.626601
Discount        12.813915
Rating           6.344593
NumReviews     -20.593725
Sales_Log      480.276925
Price_Log       33.100166
NumReviews_Log  36.586939
```

penjelasan

- Price memiliki koefisien negatif, yang berarti semakin tinggi harga, maka semakin rendah Sales.
- Discount memiliki koefisien positif, menunjukkan bahwa diskon yang lebih tinggi cenderung meningkatkan Sales.
- Sales_Log memiliki koefisien terbesar (480.2769), menandakan pengaruh yang sangat kuat terhadap prediksi.

- **Intercept Model**

```
#  Tampilkan intercept model
print(f"\nIntercept: {linear_model.intercept_:.4f}")
```

```
Intercept: -2609.7350
```

Intercept (-2609.7350) menunjukkan nilai prediksi Sales saat semua fitur bernilai nol.

- **Evaluasi Model dengan Cross-Validation (5-Fold)**

```
# 🔄 Evaluasi model dengan Cross-Validation (5-Fold)
cv_scores = cross_val_score(linear_model, X, y, cv=5, scoring="r2")

# 📊 Tampilkan hasil Cross-Validation
print(f"\nRata-rata R² Score dari Cross-Validation: {np.mean(cv_scores):.4f}")
```

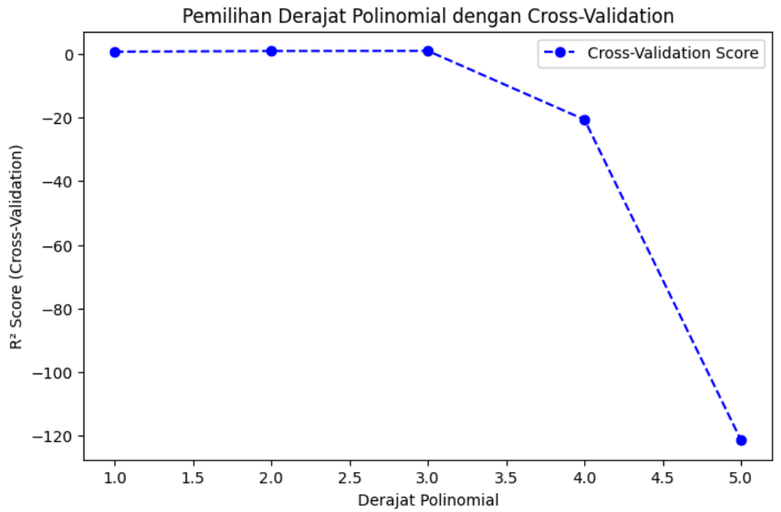
```
Rata-rata R² Score dari Cross-Validation: 0.7385
```

-R^2 Score sebesar 0.7385 menunjukkan model dapat menjelaskan 73.85% variabilitas dalam data.
-Model ini cukup baik tetapi masih bisa ditingkatkan.

- **Membangun Model Polynomial Regression**

Kita mencoba berbagai derajat polinomial (1-5) dan memilih yang terbaik berdasarkan Cross-Validation (5-Fold)

```
from sklearn.metrics import r2_score    Import "sklearn.metrics" could not be resolved from source

degrees = [1, 2, 3, 4, 5]  # Coba derajat polinomial dari 1 hingga 5
cv_scores = []

for d in degrees:
    model = make_pipeline(PolynomialFeatures(d), StandardScaler(), LinearRegression())
    scores = cross_val_score(model, X_train, y_train, cv=5, scoring="r2")  # 5-Fold CV
    cv_scores.append(np.mean(scores))  # Simpan rata-rata skor R²

# Plot hasil
plt.figure(figsize=(8,5))
plt.plot(degrees, cv_scores, marker='o', linestyle='--', color='b', label="Cross-Validation Score")
plt.xlabel("Derajat Polinomial")
plt.ylabel("R² Score (Cross-Validation)")
plt.title("Pemilihan Derajat Polinomial dengan Cross-Validation")
plt.legend()
plt.show()
```



Interpretasi Grafik:

- Derajat 1, 2, dan 3 memiliki nilai R^2 mendekati nol, menandakan model masih mampu menangkap pola dalam data.
- Derajat 4 mulai menunjukkan penurunan signifikan pada R^2, sebagai indikasi awal overfitting.
- Derajat 5 memiliki R^2 yang sangat negatif (di bawah -120), menunjukkan overfitting ekstrem dan ketidakmampuan model dalam memprediksi data secara akurat.

Grafik di atas menunjukkan hasil Cross-Validation untuk menentukan derajat terbaik pada Polynomial Regression. Sumbu X merepresentasikan derajat polinomial (1 hingga 5), sedangkan sumbu Y menunjukkan nilai R^2 Score dari 5-Fold Cross-Validation.

- **Membangun Model Polynomial Regression dengan Derajat Terbaik**

```
best_degree = degrees[np.argmax(cv_scores)]  # Pilih derajat dengan R² tertinggi

# Bangun model Polynomial Regression dengan derajat terbaik
final_model = make_pipeline(PolynomialFeatures(best_degree), StandardScaler(), LinearRegression())

# Latih model dengan data training
final_model.fit(X_train, y_train)

# Prediksi pada data uji
y_pred = final_model.predict(X_test)

# Evaluasi model
r2_test = r2_score(y_test, y_pred)

print(f"Derajat Polinomial Terbaik: {best_degree}")
print(f"R² Score pada Data Uji: {r2_test:.4f}")
```

```
Derajat Polinomial Terbaik: 3
R² Score pada Data Uji: 0.9874
```

Penjelasan :

- Derajat Polinomial Terbaik: 3
- R^2 Score pada data uji sebesar 0.9874, yang berarti model mampu menjelaskan 98,74% variabilitas dalam data.
- Model Polynomial Regression terbukti memberikan hasil yang jauh lebih baik dibandingkan dengan Linear Regression.

Evaluasi Model

- **Evaluasi Linear Regression**

```
# Prediksi menggunakan model Linear Regression
y_pred_linear = linear_model.predict(X_test)

# Hitung metrik evaluasi
mse_linear = mean_squared_error(y_test, y_pred_linear)
mae_linear = mean_absolute_error(y_test, y_pred_linear)
r2_linear = r2_score(y_test, y_pred_linear)

# Tampilkan hasil
print("🔹 Evaluasi Model Linear Regression 🔹")
print(f"Mean Squared Error (MSE): {mse_linear:.4f}")
print(f"Mean Absolute Error (MAE): {mae_linear:.4f}")
print(f"R² Score: {r2_linear:.4f}")
```

```
🔹 Evaluasi Model Linear Regression 🔹
Mean Squared Error (MSE): 71541.5582
Mean Absolute Error (MAE): 232.1312
R² Score: 0.7933
```

📉 Penjelasan Evaluasi Linear Regression

- MSE = 71.541,56 → Rata-rata kesalahan kuadrat cukup besar, menunjukkan prediksi kurang presisi.
- MAE = 232,13 → Rata-rata kesalahan absolut cukup tinggi.
- R² Score = 0,7933 → Model hanya mampu menjelaskan 79,33% variasi dalam data.

- **Evaluasi Polynomial Regression**

```
# Prediksi menggunakan model Polynomial Regression
y_pred_poly = final_model.predict(X_test)

# Hitung metrik evaluasi
mse_poly = mean_squared_error(y_test, y_pred_poly)
mae_poly = mean_absolute_error(y_test, y_pred_poly)
r2_poly = r2_score(y_test, y_pred_poly)

# Tampilkan hasil
print("\n🔹 Evaluasi Model Polynomial Regression 🔹")
print(f"Mean Squared Error (MSE): {mse_poly:.4f}")
print(f"Mean Absolute Error (MAE): {mae_poly:.4f}")
print(f"R² Score: {r2_poly:.4f}")
```

```
🔹 Evaluasi Model Polynomial Regression 🔹
Mean Squared Error (MSE): 4361.6772
Mean Absolute Error (MAE): 39.8298
R² Score: 0.9874
```

📉 Penjelasan Evaluasi polynomial Regression

- MSE = 4.361,67 → Nilai kesalahan kuadrat rata-rata jauh lebih kecil dibandingkan Linear Regression.
- MAE = 39,82 → Rata-rata error rendah, menandakan prediksi model lebih akurat.
- R² Score = 0,9874 → Model mampu menjelaskan 98,74% variasi dalam data, menunjukkan performa yang sangat baik dibandingkan Linear Regression.

Analisis Hasil

Bab ini bertujuan untuk menganalisis performa model yang telah dibangun. Fokus utama analisis mencakup:

- Interpretasi koefisien regresi dari model Linear Regression.
- Visualisasi grafik prediksi untuk melihat kecocokan antara hasil prediksi dan data asli.
- Evaluasi kelayakan model dalam melakukan prediksi nilai Sales secara akurat.

Perbandingan Kinerja Model

```
# Buat DataFrame untuk membandingkan hasil evaluasi
comparison = pd.DataFrame({
    "Model": ["Linear Regression", "Polynomial Regression"],
    "MSE": [mse_linear, mse_poly],
    "MAE": [mae_linear, mae_poly],
    "R² Score": [r2_linear, r2_poly]
})

# Tampilkan hasil
print("\n🔍 Perbandingan Kinerja Model 🔍")
print(comparison)
```

```
🔍 Perbandingan Kinerja Model 🔍
                 Model           MSE         MAE  R² Score
0    Linear Regression  71541.558228  232.131188  0.793252
1  Polynomial Regression   4361.677230   39.829751  0.987395
```

Kesimpulan Perbandingan Model

- Model Polynomial Regression derajat 3 memberikan hasil yang jauh lebih baik dibandingkan Linear Regression.
- Nilai MSE dan MAE lebih rendah, menandakan prediksi yang lebih akurat.
- R^2 Score lebih tinggi (0,9874 vs 0,7933), menunjukkan Polynomial Regression mampu menjelaskan lebih banyak variasi dalam data.

Interpretasi Koefisien Regresi Linear

```
# Menampilkan koefisien dari model Linear Regression
coefficients = pd.DataFrame(linear_model.coef_, X.columns, columns=["Koefisien"])
print("🔍 Koefisien Model Linear Regression:")
print(coefficients)

# Menampilkan intercept
print(f"\nIntercept: {linear_model.intercept_:.4f}")
```

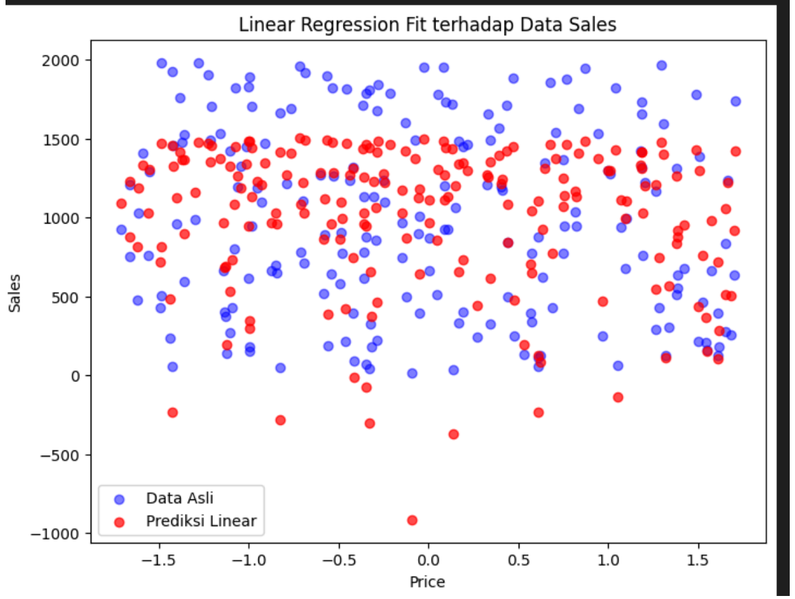
```
🔍 Koefisien Model Linear Regression:
               Koefisien
Price          -28.626601
Discount        12.813915
Rating           6.344593
NumReviews     -20.593725
Sales_Log      480.276925
Price_Log       33.100166
NumReviews_Log  36.586939

Intercept: -2609.7350
```

Penjelasan

- Price (-28.63): Harga naik → penjualan turun.
- Discount (12.81): Diskon → penjualan naik.
- Rating (6.34): Rating tinggi → sedikit meningkatkan penjualan.
- NumReviews (-20.59): Banyak ulasan → bisa jadi banyak ulasan negatif.
- Fitur Log (Sales_Log, Price_Log, NumReviews_Log): Membantu menstabilkan skala dan meningkatkan akurasi prediksi.

- **Linear Regression Fit terhadap Data Sales**

```python
# Scatter plot data asli
plt.figure(figsize=(8,6))
plt.scatter(X_test["Price"], y_test, color="blue", label="Data Asli", alpha=0.5)

# Garis prediksi model Linear Regression
plt.scatter(X_test["Price"], y_pred_linear, color="red", label="Prediksi Linear", alpha=0.7)

plt.xlabel("Price")
plt.ylabel("Sales")
plt.title("Linear Regression Fit terhadap Data Sales")
plt.legend()
plt.show()
```



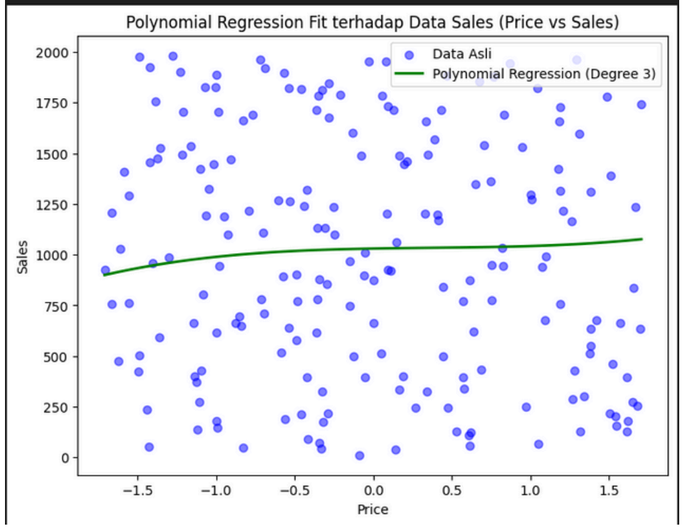
🔎 Interpretasi

- Titik merah (prediksi) tersebar di sekitar titik biru (data asli), menunjukkan model mencoba memprediksi Sales berdasarkan Price.
- Namun, banyak titik merah yang jauh dari titik biru, menandakan akurasi prediksi Linear Regression kurang baik.
- Ini menunjukkan bahwa hubungan antara Price dan Sales bersifat non-linear, sehingga tidak cukup dijelaskan dengan model linear.

- **Polynomial Regression Fit terhadap Data Sales**

```python
# Inisialisasi model Polynomial Regression dengan hanya satu fitur
poly_features_price = PolynomialFeatures(degree=best_degree)
X_train_poly_price = poly_features_price.fit_transform(X_train_price)

# Latih ulang model dengan hanya fitur Price
poly_price_model = LinearRegression()
poly_price_model.fit(X_train_poly_price, y_train)

# Buat rentang nilai Price untuk prediksi Polynomial
price_range = np.linspace(X_test_price.min(), X_test_price.max(), 100).reshape(-1, 1)
X_poly_range = poly_features_price.transform(price_range)

# Prediksi menggunakan model Polynomial Regression dengan satu fitur
y_poly_pred = poly_price_model.predict(X_poly_range)

# Plot hasil Polynomial Regression
plt.figure(figsize=(8,6))
plt.scatter(X_test_price, y_test, color="blue", label="Data Asli", alpha=0.5)
plt.plot(price_range, y_poly_pred, color="green", label=f"Polynomial Regression (Degree {best_degree})", l

plt.xlabel("Price")
plt.ylabel("Sales")
plt.title("Polynomial Regression Fit terhadap Data Sales (Price vs Sales)")
plt.legend()
plt.show()
```

Python



🔎 Interpretasi

- Dibanding Linear Regression, model ini lebih baik dalam menangkap pola non-linear.
- Tapi grafik menunjukkan kurvanya masih datar, jadi hubungan antara Price dan Sales tetap lemah.
- Artinya, Price saja belum cukup untuk memprediksi Sales. Perlu mempertimbangkan fitur lain seperti Discount, Rating, dan NumReviews.

Kesimpulan

Setelah melalui serangkaian tahapan, mulai dari Pemahaman Dataset, Eksplorasi Data dan Pra-pemrosesan, Implementasi Model, Evaluasi Model, hingga Analisis Hasil, kita dapat menyimpulkan bahwa model Polynomial Regression mampu menangkap pola hubungan antara variabel dengan lebih baik dibandingkan Linear Regression dalam memprediksi Sales. Hal ini terlihat dari hasil prediksi yang lebih sesuai dengan distribusi data, menunjukkan bahwa pendekatan non-linear lebih efektif dalam merepresentasikan hubungan antara Price dan Sales. Dengan demikian, penggunaan Polynomial Regression menjadi pilihan yang lebih optimal untuk meningkatkan akurasi prediksi dalam konteks ini.