Regularisasi dalam Regresi

Regularisasi adalah teknik yang digunakan dalam machine learning untuk mencegah overfitting. Overfitting terjadi ketika model terlalu kompleks dan menyesuaikan diri terlalu baik dengan data latih, sehingga tidak dapat menggeneralisasi dengan baik pada data baru. Dalam regresi, overfitting dapat menyebabkan model memiliki koefisien regresi yang sangat besar dan tidak stabil.

Regularisasi bekerja dengan menambahkan penalti pada fungsi loss model. Penalti ini menghukum model yang terlalu kompleks, sehingga mendorong model untuk memilih solusi yang lebih sederhana dan lebih umum.

# Jenis-jenis Regularisasi:

1. **L1 Regularization (LASSO):**

* Menambahkan penalti berdasarkan nilai absolut dari koefisien regresi.
* Dapat menghasilkan koefisien regresi yang bernilai nol (fitur selection).

1. **L2 Regularization (Ridge):**

* Menambahkan penalti berdasarkan kuadrat dari koefisien regresi.
* Mencegah koefisien regresi menjadi terlalu besar, tetapi tidak menghasilkan koefisien regresi nol.

1. **Elastic Net:**

* Kombinasi dari L1 dan L2 regularization.
* Memberikan fleksibilitas untuk mengatur keseimbangan antara fitur selection dan pencegahan koefisien besar.

# Penerapan Regularisasi dengan Scikit-Learn

Scikit-learn menyediakan beberapa model regresi yang sudah dilengkapi dengan regularisasi, seperti:

* **Ridge:** Regresi linear dengan L2 regularization.
* **Lasso:** Regresi linear dengan L1 regularization.
* **ElasticNet:** Regresi linear dengan kombinasi L1 dan L2 regularization.

**Tugas Hari 21**

Lanjutan dengan dataset “Advertising” dan model regresi linear yang telah Anda buat.

**Tugas:**

1. **Membuat Model Ridge Regression:**

* Buat model Ridge Regression menggunakan scikit-learn dengan nilai alpha (parameter regularisasi) = 1.0
* Latih model pada data latih.
* Evaluasi kinerja model pada data uji menggunakan MAE, MSE, dan RMSE.

1. **Membuat Model Lasso Regression:**

* Buat model Lasso Regression menggunakan scikit-learn dengan nilai alpha = 1.0.
* Latih model pada data latih.
* Evaluasi kinerja model pada data uji menggunakan MAE, MSE, dan RMSE.

1. **Membuat Model Elastic Net Regression:**

* Buat model Elastic Net Regression menggunakan scikit-learn dengan nilai alpha = 1.0 dan |1\_ratio = 0.5.
* Latih model pada data latih.
* Evaluasi kinerja model pada data uji menggunakan MAE, MSE, dan RMSE.

1. **Bandingkan Kinerja:**

Bandingkan kinerja ketiga model tersebut (Ridge, Lasso, dan Elastic Net) dengan model regresi linear yang tanpa regularisasi. Model mana yang memiliki kinerja terbaik pada data uji? Jelaskan alasan Anda.

**Contoh Kode (Scikit-Learn)**

from sklearn.linear\_model import Ridge, Lasso, ElasticNet

# ... (kode untuk membaca dan mempersiapkan dataset Advertising)

# Ridge Regression

ridge\_model = Ridge(alpha=1.0)

ridge\_model.fit(X\_train, y\_train)

# Evaluasi...

# Lasso Regression

lasso\_model = Lasso(alpha=1.0)

lasso\_model.fit(X\_train, y\_train)

# Evaluasi...

# Elastic Net Regression

elastic\_model = ElasticNet(alpha=1.0, l1\_ratio=0.5)

elastic\_model.fit(X\_train, y\_train)

# Evaluasi...

**Catatan:**

Anda dapat mencoba nilai alpha yang berbeda untuk melihat bagaimana hal tersebut mempengaruhi kinerja model.

**Selamat Mengerjakan Tugas! 😣**

**Tugas**

1. **Membuat Model Ridge Regression:**

**ridge\_model = Ridge(alpha=1.0)**

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

**# EVALUASI MSE, MAE, R2\_SCORE, RMSE**

**# Melakukan prediksi pada data uji**

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

**# Menghitung Mean Squared Error (MSE)**

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

**# Menghitung Mean Absolute Error (MAE)**

**mae = mean\_absolute\_error(y\_test, y\_pred)**

**# Menghitung R-squared (R²)**

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

**# RMSE**

**rmse = np.sqrt(mse)**

**print(f'Mean Squared Error (MSE): {mse}')**

**print(f'Mean Absolute Error (MAE): {mae}')**

**print(f'R-squared (R²): {r2}')**

**print(f'RMSE (RMSE): {rmse}')**

1. **Membuat Model Lasso Regression:**

lasso\_model = Lasso(alpha=1.0)

lasso\_model.fit(X\_train, y\_train)

# EVALUASI MSE,MAE, RMSE,R2\_SCORE

# Melakukan prediksi pada data uji

y\_pred = lasso\_model.predict(X\_test)

# Menghitung Mean Squared Error (MSE)

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

# Menghitung Mean Absolute Error (MAE)

mae = mean\_absolute\_error(y\_test, y\_pred)

# Menghitung R-squared (R²)

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

# RMSE

rmse = np.sqrt(mse)

print(f'Mean Squared Error (MSE): {mse}')

print(f'Mean Absolute Error (MAE): {mae}')

print(f'R-squared (R²): {r2}')

print(f'RMSE (RMSE): {rmse}')

1. **Membuat Model Elastic Net Regression:**

elastic\_model = ElasticNet(alpha=1.0, l1\_ratio=0.5)

elastic\_model.fit(X\_train, y\_train)

# EVALUASI MSE, MAE, RMSE,R2\_SCORE

# Melakukan prediksi pada data uji

y\_pred = elastic\_model.predict(X\_test)

# Menghitung Mean Squared Error (MSE)

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

# Menghitung Mean Absolute Error (MAE)

mae = mean\_absolute\_error(y\_test, y\_pred)

# Menghitung R-squared (R²)

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

# RMSE

rmse = np.sqrt(mse)

print(f'Mean Squared Error (MSE): {mse}')

print(f'Mean Absolute Error (MAE): {mae}')

print(f'R-squared (R²): {r2}')

print(f'RMSE (RMSE): {rmse}')

1. **Bandingkan Kinerja:**

**\*\*Ridge Regression\*\***

1. Mean Squared Error (MSE): 8.322656831326814

2. Mean Absolute Error (MAE): 2.312856304829525

3. R-squared (R²): 0.9990104334651879

4. RMSE (RMSE): 2.8849015288787268

**\*\*Lasso Regression\*\***

1. Mean Squared Error (MSE): 8.310689468545958

2. Mean Absolute Error (MAE): 2.311257176471573

3. R-squared (R²): 0.9990118563884152

4. RMSE (RMSE): 2.882826645593862

**\*\*Elastic Net\*\***

1. Mean Squared Error (MSE): 8.310689468545958

2. Mean Absolute Error (MAE): 2.311257176471573

3. R-squared (R²): 0.9990118563884152

4. RMSE (RMSE): 2.882826645593862

**Analisis Perbandingan Kinerja:**

| Model | MAE | MSE | RMSE | R-squared |
| --- | --- | --- | --- | --- |
| Linear Regression | 2.24 | 7.98 | 2.82 | 0.9990 |
| Ridge (alpha=1.0) | 2.31 | 8.32 | 2.88 | 0.9990 |
| Lasso (alpha=1.0) | 2.31 | 8.31 | 2.88 | 0.9990 |
| Elastic Net (alpha=1.0, l1\_ratio=0.5) | 2.31 | 8.31 | 2.88 | 0.9990 |

**Interpretasi Hasil:**

* **Linear Regression (tanpa regularisasi):** Model ini memiliki MAE, MSE, dan RMSE yang sedikit lebih rendah daripada model-model dengan regularisasi, tetapi perbedaannya sangat kecil. Ini menunjukkan bahwa model tanpa regularisasi sudah cukup baik dan mungkin tidak mengalami overfitting yang signifikan.
* **Ridge, Lasso, dan Elastic Net:** Ketiga model ini memiliki kinerja yang sangat mirip satu sama lain. Hal ini bisa disebabkan oleh beberapa faktor, seperti:
  + **Nilai alpha yang tinggi:** Nilai alpha = 1.0 yang Anda gunakan mungkin terlalu tinggi, sehingga efek regularisasi menjadi terlalu kuat. Anda bisa mencoba nilai alpha yang lebih kecil.
  + **Fitur yang tidak terlalu berkorelasi:** Mungkin tidak ada multikolinearitas yang signifikan dalam dataset Anda, sehingga LASSO (L1 regularization) tidak memberikan banyak keuntungan dalam pemilihan fitur.
  + **Dataset yang cukup besar:** Dataset "Advertising" mungkin cukup besar sehingga overfitting tidak menjadi masalah utama, bahkan tanpa regularisasi.

**Kesimpulan:**

Dalam kasus ini, model regresi linear tanpa regularisasi tampaknya memberikan kinerja yang cukup baik. Namun, jika Anda ingin lebih yakin tentang pemilihan model, Anda bisa mencoba beberapa hal berikut:

* **Tuning Hyperparameter:** Lakukan tuning hyperparameter (terutama alpha) untuk menemukan nilai yang optimal untuk masing-masing model regularisasi.
* **Cross-Validation:** Lakukan cross-validation untuk mendapatkan estimasi kinerja yang lebih robust.
* **Analisis Lebih Lanjut:** Lakukan analisis lebih lanjut untuk memahami mengapa model-model regularisasi tidak memberikan peningkatan kinerja yang signifikan dalam kasus ini.

**Catatan:**

* Pemilihan model terbaik tidak selalu mudah dan tergantung pada banyak faktor. Anda perlu mempertimbangkan tujuan analisis Anda, karakteristik dataset, dan metrik evaluasi yang Anda gunakan.