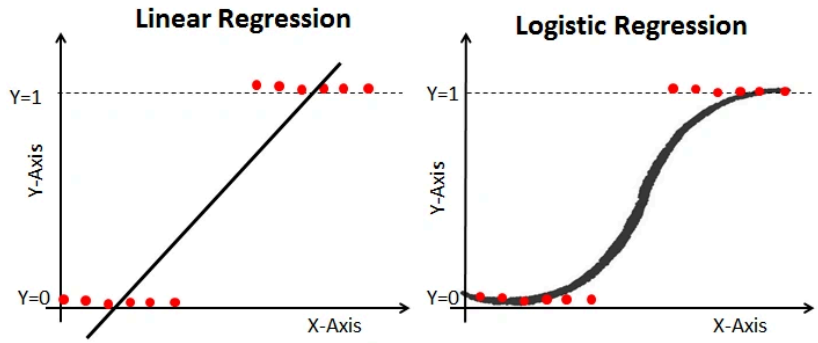
Logistic Regression

Cara Kerja



Sumber:
https://www.analyticsvidhya.com/blog/2021/08/conceptual-understanding-of-logistic-regression-for-data-science-beginners/

Diberikan sebuah model persamaan linear:

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

atau dalam persamaan matriksnya yakni:

$$z = WX + b$$

dengan $b = \beta_0$ adalah *bias*, W atau β_1, β_2, ..., β_n adalah *weights*, dan X atau $x_1, x_2, \ldots, x_n$ adalah fitur data input. Diberikan juga sebuah fungsi sigmoid:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Secara sederhana, model logistic regression melakukan prediksi dengan memasukkan input data ke dalam fungsi komposisi model linear dan sigmoid:

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

Dalam proses *learning*-nya, model logistic regression mengubah nilai weights dan bias pada model linear. Perubahan weight dan bias pada model linear akan mengubah bentuk kurva sigmoid pada fungsi komposisi akhir. Untuk mencari weights dan bias yang optimal, digunakan metode gradient descent.

Loss function yang digunakan untuk logistic regression untuk binary classification adalah adalah log loss (kasus khusus dari cross entropy yang hanya menangani dua jenis kelas). Untuk mengetahui seberapa besar kontribusi dari setiap weight dan bias terhadap error, digunakan

gradien dari loss function terhadap setiap weights dan bias. Gradien ini dihitung pada setiap iterasi gradient descent. Setelah mengetahui nilai gradien tersebut, weights dan bias diperbarui dengan cara menguranginya dengan gradien tersebut dikalikan dengan learning rate.

$$\text{logloss}(a, y) = -y \log a - (1 - y) \log(1 - a)$$

$$a = \text{sigmoid}(w^{\intercal} x) \equiv \frac{1}{1 + e^{-w^{\intercal} x}}$$

$$\frac{\partial \text{logloss}}{\partial w} = (a - y)x$$

$$w := w - \alpha(a - y)x$$

Evaluasi Model

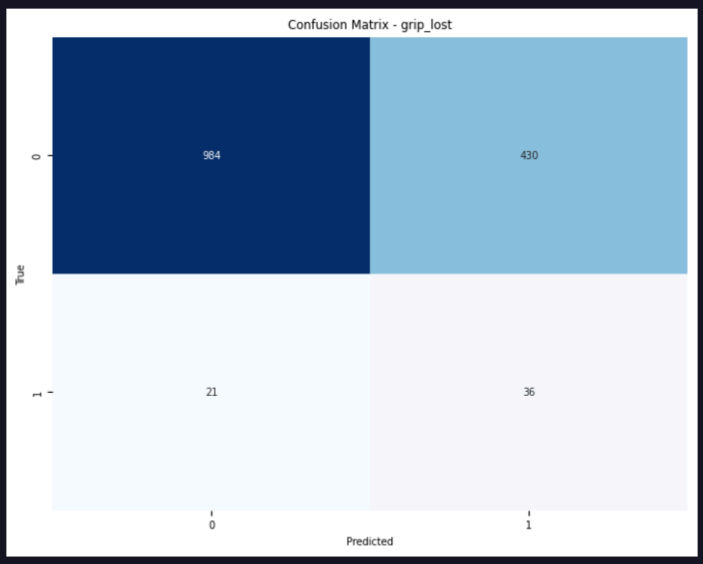
Evaluasi	Hasil
Model from Scratch	

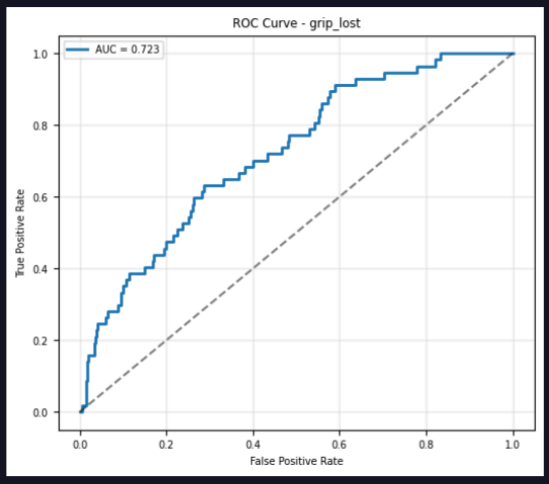
Hold-out validation untuk grip_lost

```
============================================================
Hold-Out Validation for grip_lost:
============================================================
Training time: 0.008997678756713867 seconds
Prediction time: 0.0019991397857666016 seconds
              precision    recall  f1-score   support

           0       0.98      0.70      0.81      1414
           1       0.08      0.63      0.14        57

    accuracy                           0.69      1471
   macro avg       0.53      0.66      0.48      1471
weighted avg       0.94      0.69      0.79      1471
```





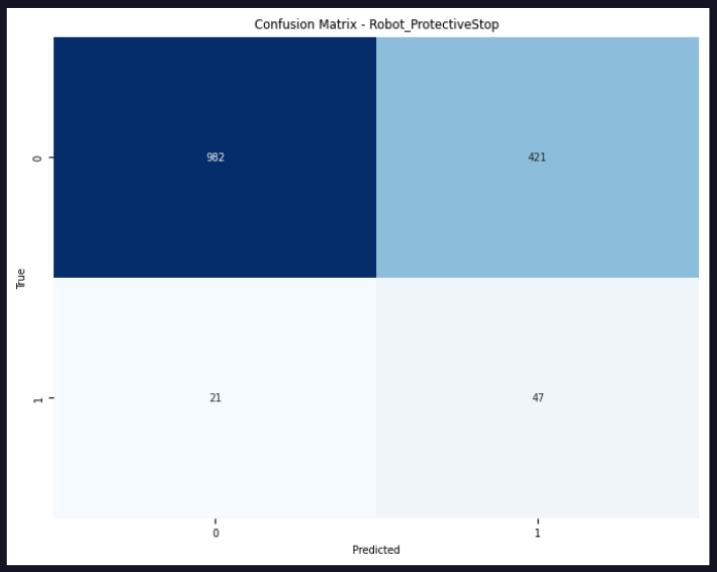

```
AUC-ROC: 0.7229
```

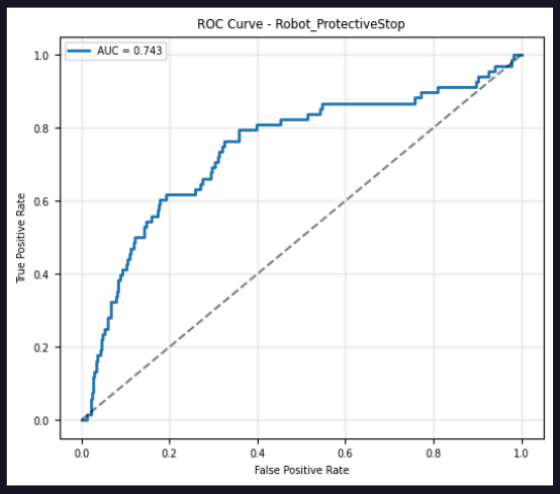
Hold out validation untuk Robot_ProtectiveStop

```
============================================================
Hold-Out Validation for Robot_ProtectiveStop:
============================================================
Training time: 0.008998394012451172 seconds
Prediction time: 0.001999378204345703 seconds
              precision    recall  f1-score   support

           0       0.98      0.70      0.82      1403
           1       0.10      0.69      0.18        68

    accuracy                           0.70      1471
   macro avg       0.54      0.70      0.50      1471
weighted avg       0.94      0.70      0.79      1471
```






```
AUC-ROC: 0.7434
```

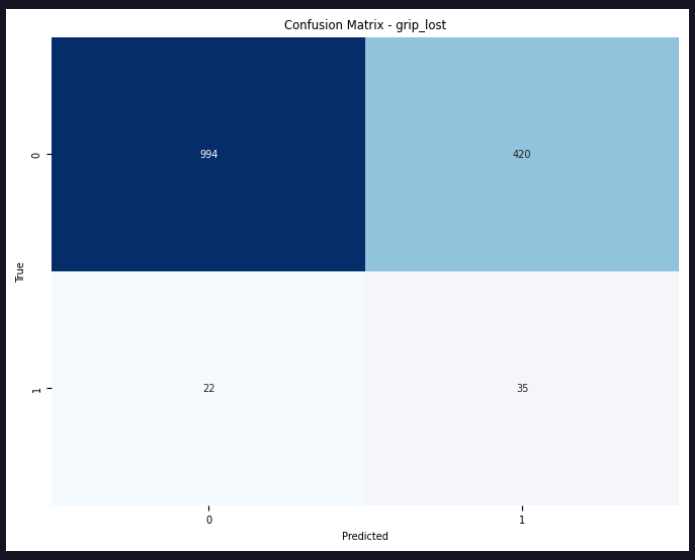
K-fold cross validation untuk grip_lost	``` == Cross Validation for grip_lost: == Model Performance (5-Fold Cross Validation): fit_time: [0.02091694 0.01001406 0.01321912 0.01087856 0.01200509] score_time: [0.00599051 0.00699115 0.00600767 0.00700021 0.00599933] test_precision: [0.07725322 0.08230453 0.06593407 0.0661157 0.06436782] Average test_precision: 0.07 test_recall: [0.63157895 0.71428571 0.68181818 0.76190476 0.63636364] Average test_recall: 0.69 test_f1: [0.1376673 0.14760148 0.12024048 0.121673 0.11691023] Average test_f1: 0.13 ```
K-fold cross validation untuk Robot_ProtectiveStop	``` == Cross Validation for Robot_ProtectiveStop: == Model Performance (5-Fold Cross Validation): fit_time: [0.0099988 0.00999856 0.00999975 0.01303911 0.01499939] score_time: [0.00600171 0.00600362 0.00599957 0.00500202 0.00600386] test_precision: [0.10042735 0.07302231 0.10141988 0.07256236 0.08210526] Average test_precision: 0.09 test_recall: [0.69117647 0.67924528 0.83333333 0.68085106 0.78] Average test_recall: 0.73 test_f1: [0.17537313 0.13186813 0.18083183 0.13114754 0.14857143] Average test_f1: 0.15 ```
Model Scikit-Learn	

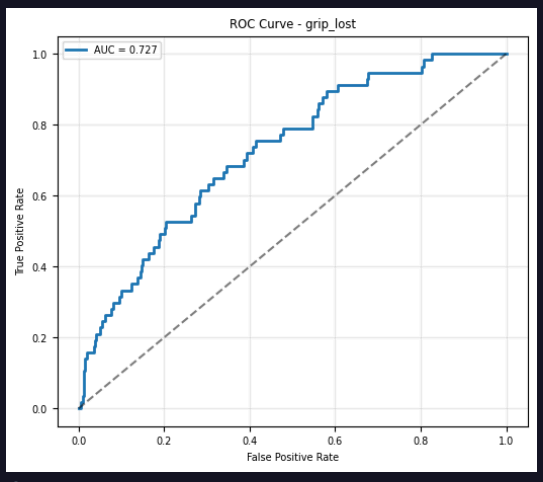
Hold-out validation untuk grip_lost

```
============================================================
Hold-Out Validation for grip_lost:
============================================================
Training time: 0.016236543655395508 seconds
Prediction time: 0.0010020732879638672 seconds
              precision    recall  f1-score   support

           0       0.98      0.70      0.82      1414
           1       0.08      0.61      0.14        57

    accuracy                           0.70      1471
   macro avg       0.53      0.66      0.48      1471
weighted avg       0.94      0.70      0.79      1471
```





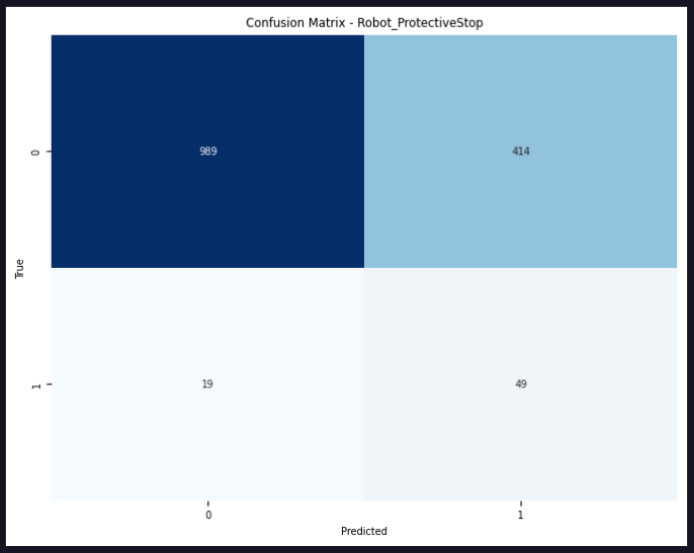

```
AUC-ROC: 0.7271
```

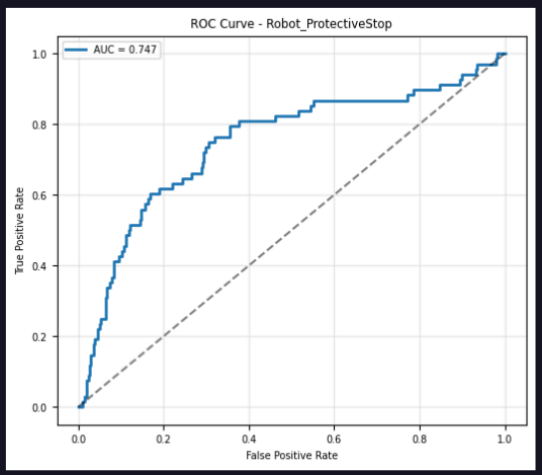
Hold out validation untuk Robot_ProtectiveStop

```
============================================================
Hold-Out Validation for Robot_ProtectiveStop:
============================================================
Training time: 0.015000581741333008 seconds
Prediction time: 0.0009996891021728516 seconds
              precision    recall  f1-score   support

           0       0.98      0.70      0.82      1403
           1       0.11      0.72      0.18        68

    accuracy                           0.71      1471
   macro avg       0.54      0.71      0.50      1471
weighted avg       0.94      0.71      0.79      1471
```






```
AUC-ROC: 0.7470
```

K-fold cross validation untuk grip_lost	``` == Cross Validation for grip_lost: == Model Performance (5-Fold Cross Validation): fit_time: [0.01751661 0.01575351 0.01452184 0.01900291 0.023] score_time: [0.00300002 0.00399899 0.00200129 0.00550628 0.00701213] test_precision: [0.07692308 0.08613445 0.06651885 0.06694561 0.06839623] Average test_precision: 0.07 test_recall: [0.61403509 0.73214286 0.68181818 0.76190476 0.65909091] Average test_recall: 0.69 test_f1: [0.13671875 0.15413534 0.12121212 0.12307692 0.12393162] Average test_f1: 0.13 ```
K-fold cross validation untuk Robot_ProtectiveStop	``` == Cross Validation for Robot_ProtectiveStop: == Model Performance (5-Fold Cross Validation): fit_time: [0.01851535 0.01599932 0.01340389 0.02100015 0.01801085] score_time: [0.00299811 0.00300384 0.00300169 0.00500298 0.00651455] test_precision: [0.10583153 0.07438017 0.09958506 0.07517084 0.08385744] Average test_precision: 0.09 test_recall: [0.72058824 0.67924528 0.8 0.70212766 0.8] Average test_recall: 0.74 test_f1: [0.18455744 0.13407821 0.17712177 0.13580247 0.15180266] Average test_f1: 0.16 ```

Implementasi logistic regression *from scratch* ini memberikan hasil yang berbeda daripada implementasi scikit-learn. Model yang dibuat dari awal ini menghasilkan metrik recall yang secara rata-rata sedikit lebih rendah. Hal yang mungkin terjadi adalah ketidaksesuaian implementasi algoritma secara menyeluruh, mengingat bahwa implementasi logistic regression scikit-learn memiliki banyak parameter seperti solver, random_state, dan tol.

Improvement

Secara implementasi, karena model ini digunakan untuk klasifikasi biner, model diimplementasikan menggunakan loss function log loss (kasus khusus dari cross entropy). Model dapat diimplementasikan lebih lanjut untuk menerapkan cross entropy agar dapat menangani klasifikasi multiclass. Feature engineering, feature selection, atau dimensionality reduction yang tepat dapat dikaji lebih lanjut untuk meningkatkan kemampuan prediksi model.