Modul Ajar

Kecerdasan Buatan Lanjut

Evaluasi Model *Machine Learning*

CPL

1. Mampu menggunakan dan mengoptimalisasi teknik akuisisi data, analisis data, pengolahan data untuk menyelesaikan masalah penelitian atau industry
2. mampu menguasai konsep dan pengetahuan teknik akuisisi data, analisis data, pengolahan data untuk menyelesaikan masalah

CPMK

1. Mahasiswa mampu menerapkan metode analisis data
2. Mahasiswa mampu menerapkan metode evaluasi data
3. Mahasiswa menjelaskan konsep teknik akuisisi data
4. Mahasiswa mampu menerapkan teknik pengolahan data untuk menyelesaikan masalah

Sub CPMK

Mahasiswa mampu menjelaskan machine learning dan evaluasinya

Indikator Keberhasilan

Mahasiswa dapat menjelaskan metode evaluasi model, seperti confusion matrix, precision, recall, F1-score, ROC-AUC, dan cross-validation.

Pokok Bahasan

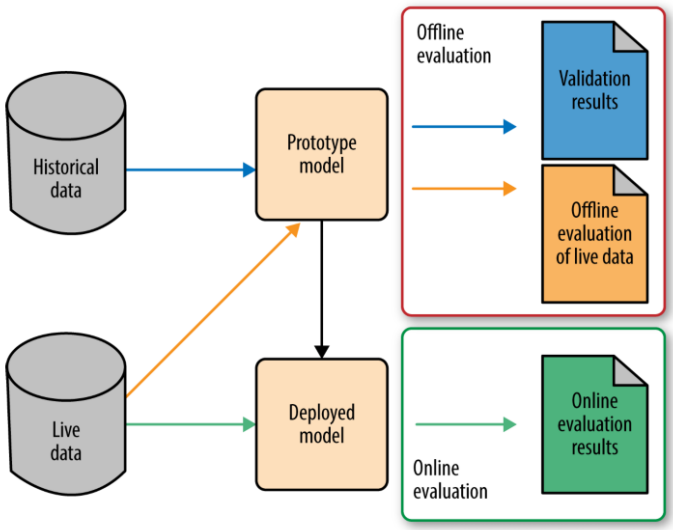
1. Evaluasi *Machine Learning Model*
2. *Machine Learning Workflow*
3. Metrik Evaluasi
4. Mekanisme Evaluasi Secara *Offline*
5. Mekanisme Evaluasi Secara Daring

1. **Evaluasi *Machine Learning Model***

Evaluasi model merupakan proses menganalisa kinerja dari model machine learning menggunakan beberapa metrik evaluasi. Tahapan ini penting dalam siklus pengembangan *machine learning* karena menentukan seberapa baik model yang dibangun mampu menyelesaikan permasalahan pada data baru. Melatih model dapat dianalogikan seperti mengajar seorang siswa. Evaluasi model ibarat ujian untuk melihat apakah mereka benar-benar memahami materi – atau hanya menghafal jawaban. Hasil evaluasi performa sebuah model, dapat digunakan sebagai acuan untuk melakukan optimasi/ perbaikan kinerja model.

2. **Machine Learning *Workflow***

Machine learning workflow merupakan serangkaian Langkah sistematis yang dilakukan untuk membangun, melatih, mengevaluasi dan menerapkan sebuah model Machine learning. Workflow ini bertujuan agar pengembangan model. Workflow dalam pengembangan Model Machine Learning, dapat ditunjukkan oleh gambar 1 berikut:



Gambar 1. *Workflow* Pengembangan & Evaluasi *Machine Learning Model*
Sumber:

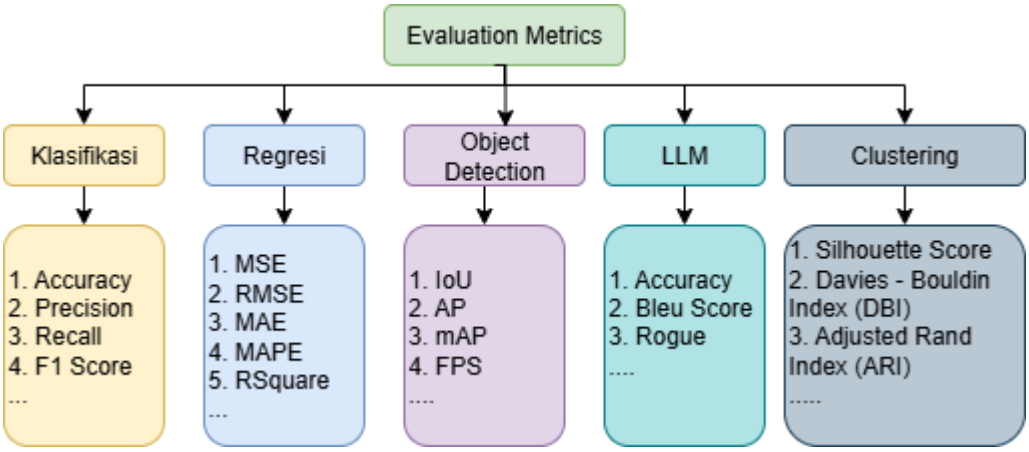
Pada tahap awal, data lampau dimanfaatkan untuk membangun dan menguji prototype model. Evaluasi dilakukan secara *offline* dengan dua pendekatan. Pertama, validasi terhadap data historis menghasilkan validation results, yang merefleksikan kemampuan model dalam mengenali pola serta hubungan pada data lampau. Kedua, prototipe juga dievaluasi menggunakan live data dalam konteks offline guna menilai sejauh mana model tetap robust dan adaptif ketika dihadapkan pada data aktual yang lebih dinamis. Apabila hasil evaluasi menunjukkan kinerja yang memadai, model siap untuk di deploy. Pada fase ini, model mulai menerima aliran live data secara langsung, dan performanya diuji melalui online evaluation, yang menghasilkan online evaluation results. Proses ini menekankan bahwa evaluasi model tidak hanya berhenti pada validasi berbasis data historis, tetapi juga mencakup pengujian terhadap data aktual, baik secara offline maupun online, untuk memastikan reliabilitas, relevansi, dan kemampuan adaptasi model terhadap kondisi lingkungan yang terus berubah.

3. Metrik Evaluasi

Banyak sekali metrik evaluasi yang dapat digunakan untuk mengukur performa dari sebuah model. Pemilihan metrik yang digunakan dalam evaluasi model sangat bergantung kepada:

- Jenis masalah (klasifikasi, regresi, clustering, dsb)
- Tujuan utama (efisiensi, efektitas, keseimbangan kelas, dsb)
- Konteks masalah (medis, e-commerce, finance, dll)

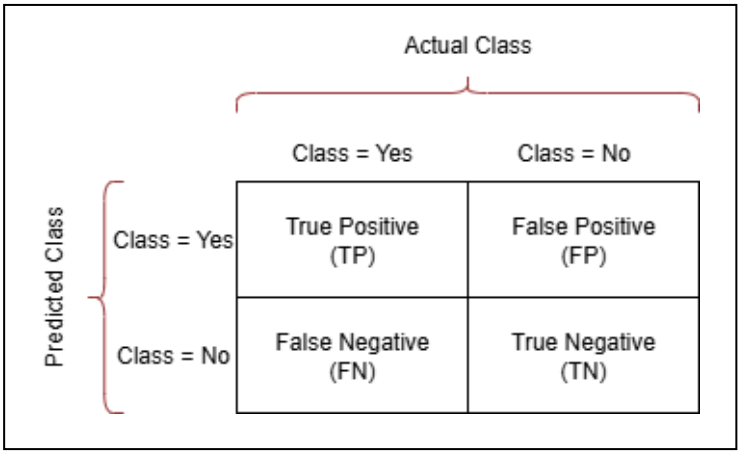
Beberapa contoh metrik yang dapat digunakan untuk melakukan evaluasi model, berdasarkan jenis masalah dapat dilihat pada gambar 2 berikut:



Gambar 2. Metrik Evaluasi Berdasarkan Jenis Masalah

a. Metric Evaluasi Pada Klasifikasi

Metrik dasar dalam evaluasi klasifikasi berawal dari confusion matrix, yaitu tabel yang menunjukkan perbandingan antara hasil prediksi dengan label actual, seperti yang diperlihatkan gambar 3. Dari confusion matrix inilah lahir berbagai metrik turunan, seperti accuracy, precision, recall dan F1 Score.



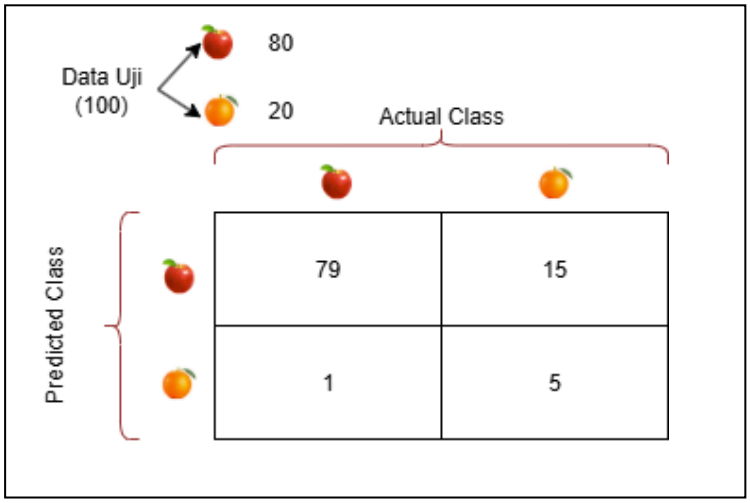
Gambar 3. Confusion Matric Binary Classification

Akurasi (*Accuracy*)

Akurasi merupakan metrik yang paling popular dalam klasifikasi. Akurasi menunjukkan proporsi prediksi yang benar terhadap keseluruhan data. Akurasi dapat dihitung dengan persamaan 1 berikut:

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (1)$$

Meskipun, metrik akurasi sering digunakan dalam mengukur performa klasifikasi, namun akurasi kurang efektif Ketika distribusi kelas tidak seimbang. Dalam *imbalance dataset*, model dapat terlihat memiliki akurasi yang tinggi meskipun gagal mengenali kelas minoritas, lihat contoh kasus pada gambar 4 di bawah ini:



Gambar 4. Confusion Matrix Jeruk vs Apel

Pada kasus tersebut model memili akurasi sebesar 84 %, sehingga kita anggap memili performa yang cukup baik. Namun sejatinya model gagal dalam mengenali kelas jeruk. Ketepatan model dalam mengenali kelas jeruk, hanya sebesar $\frac{5}{20}$ atau 25%.

Presisi (*Precision*)

Metrik Precision dapat digunakan untuk mengukur seberapa tepat model dalam memberikan prediksi positif. Nilai prediksi yang tinggi menunjuukan bahwa model jarang salah memberikan label positif. Metrik presisi dapat dihitung dengan menggunakan persamaan 2 berikut ini:

$$precision = \frac{TP}{TP + FP} \quad (2)$$

Metrik *precision* ini sangat penting pada kasus- kasus Dimana *false positive* beresiko tinggi, misalnya pada klasifikasi email spam dan bukan spam. Akan sangat fatal jika ada email bukan spam yang diidentifikasi sebagai spam. Email akan masuk trash dan tidak akan pernah terbaca.

Recall (Sensitivity atau True Positive Rate)

Metrik Recall digunakan untuk mengukur seberapa baik model menangkap seluruh data positif yang ada. Nilai recall cocok digunakan untuk kasus-kasus yang ingin meminimalisir false negative. Misalnya kasus-kasus di bidang medis: kesalahan diagnosa dari pasien yang sesungguhnya sakit, namun terdeteksi tidak sakit, akan berakibat fatal. Cara menghitung nilai recall ditunjukkan oleh persamaan 3 berikut:

$$recall = \frac{TP}{TP + FN} \tag{3}$$

F1-Score

Metrik F1 Score merupakan rata-rata harmoni dari precision dan recall. F1 Score lebih representative dibandingkan melihat prescision dan recall secara terpisah. Sehigga metrik f1 score akan cocok digunakan pada kondisi data yang tidak seimbang(imbalance). Metrik f1 score dihitung dengan persamaan 4 berikut:

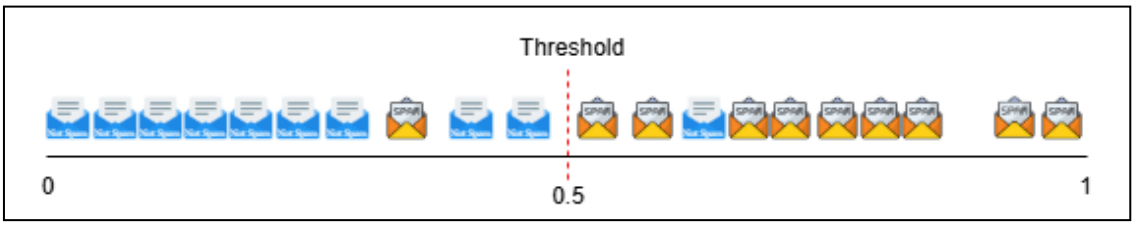
$$f1 - Score = 2 \times \frac{precision \times recall}{precision + recall} \tag{4}$$

Kurva ROC - AUC

ROC merupakan singkatan dari Receiver Operating Characteristic. Metrik ini digunakan untuk mengevaluasi klasifikasi, khusunya binary classification (2 kelas). ROC merupakan kurva yang menggambarkan hubungan antara True Positive Rate (TPR) dengan False Positive Rate (FPR) dengan nilai ambang batas (threshold) klasifikasi yang bervariasi. Nilai TPR merupakan nilai Recall atau Sensitivity, sedangkan nilai FPR di hitung dengan persamaan 5 berikut:

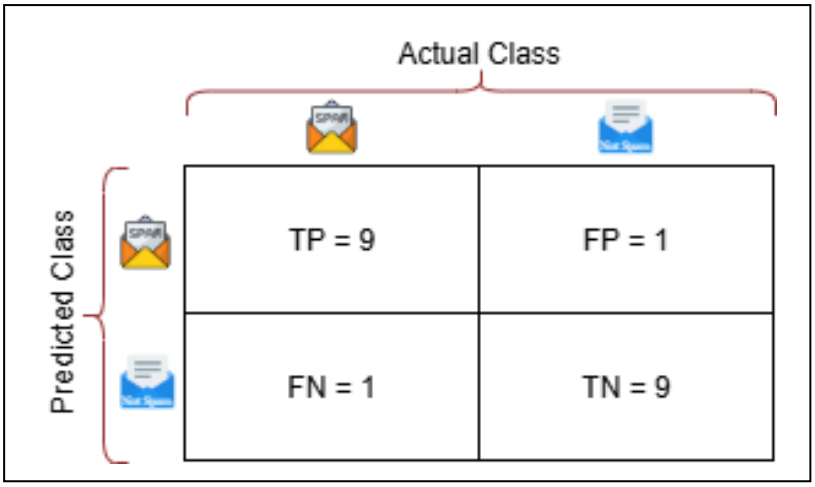
$$FPR = \frac{FP}{FP + TN} \tag{5}$$

Nilai TPR akan memotret model dalam melakukan identifikasi kasus positif dengan benar. Seadangkan FPR digunakan untuk mengetahui sejauh mana model memprediksi kasus negative sebagai positif. Perhatikan gambar 5 berikut:



Gambar 5. Klasifikasi Email Spam dengan *Threshold*

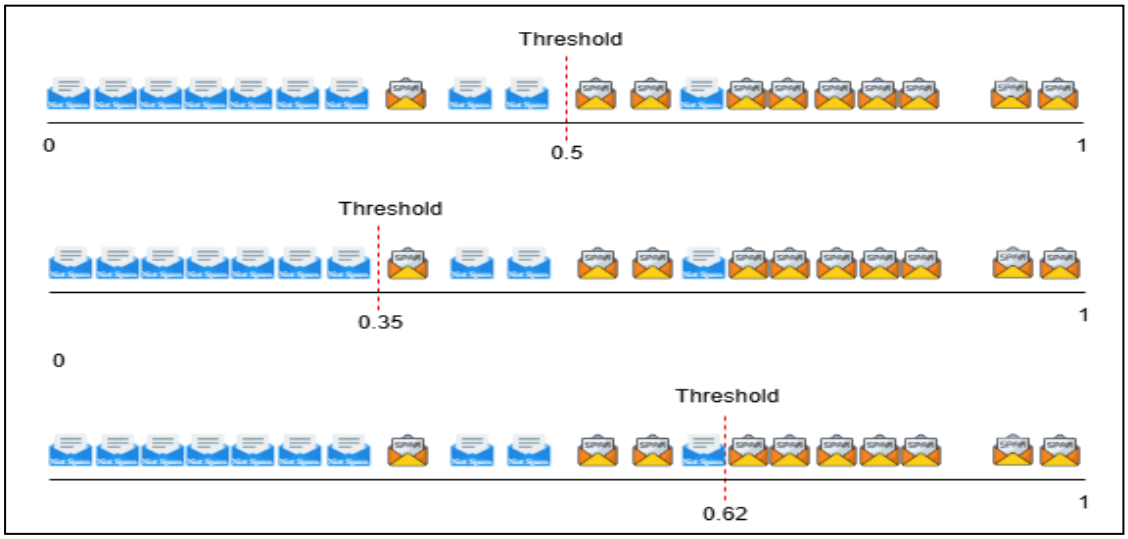
Berdasarkan gambar tersebut di atas, model dapat memisahkan kelas spam dan bukan spam menggunakan threshold (nilai ambang) 0.5. Jika $data \geq threshold$ maka akan diidentifikasi sebagai email spam, begitu pula sebaliknya, jika $data < threshold$ maka akan diidentifikasi sebagai bukan spam (ham). Nilai confusion matrix dari model tersebut ditunjukkan oleh gambar 6 di bawah ini:



Gambar 6. *Confusion Matrix* Klasifikasi Email

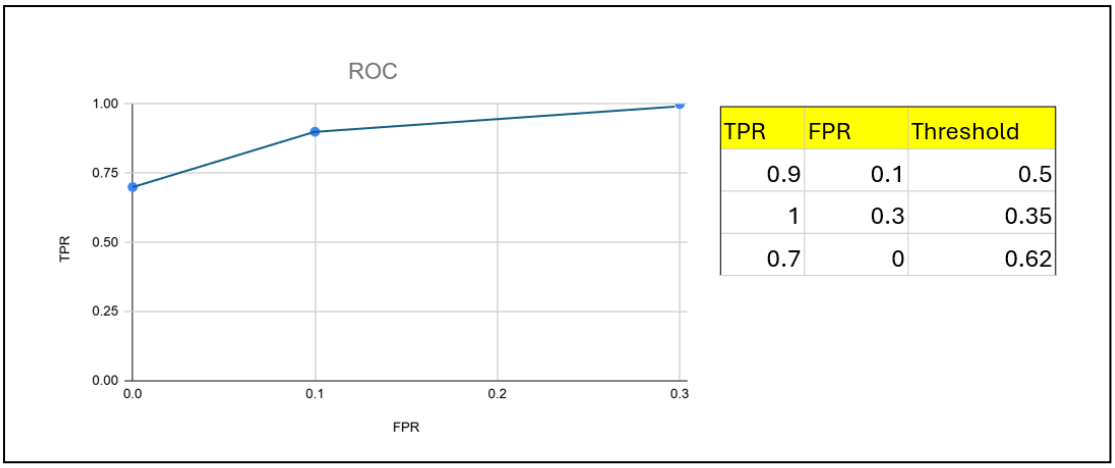
Dari gambar diatas dapat kita hitung nilai TPR dan FPRnya. Nilai TPR yang dihasilkan oleh model tersebut adalah $\frac{TP}{TP+FN} = \frac{9}{9+1} = 0.9$, sedangkan nilai FPRnya Adalah $\frac{FP}{FP+TN} = \frac{1}{1+9} = 0.1$

Andai kita mengubah nilai threshold, maka *confusion matrix* yang dihasilkan oleh model juga akan berubah. Perhatikan perubahan *threshold* seperti yang ditunjukkan oleh gambar 7 berikut:



Gambar 7. Perubahan Threshold

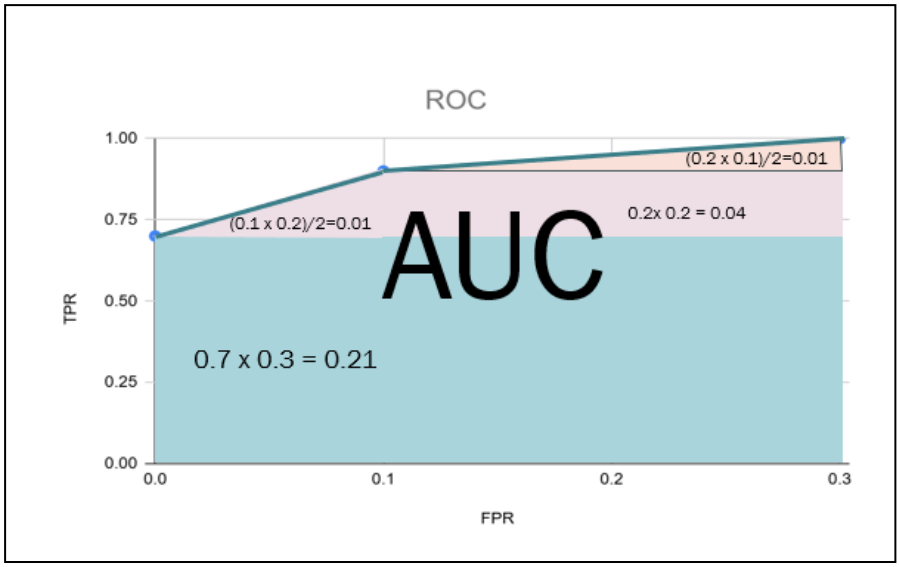
Perubahan *threshold* akan mempengaruhi *confusion matrix*. *Confusion matrix* akan mempengaruhi nilai TPR dan FPR. Oleh karena itu, kita dapat memetakan nilai TPR dan FPR dengan nilai *threshold* yang bervariasi ini ke dalam kurva ROC. Pada ROC, setiap titik akan mewakili satu *threshold* tertentu. Makin Dekat kurva ROC ke sudut kiri atas, maka semakin baik performa sebuah model. Contoh kurva ROC dengan 4 *threshold* diperlihatkan oleh gambar 8 berikut ini:



TPR	FPR	Threshold
0.9	0.1	0.5
1	0.3	0.35
0.7	0	0.62

Gambar 8. Kurva ROC

Area Under the Curve (AUC) merupakan nilai yang dihitung dari luas area yang berada di bawah kurva ROC. Nilai AUC berada dalam rentang 0 dan 1. Nilai AUC dapat mengukur seberapa baik model dapat membedakan positif dan negative tanpa memandang nilai *threshold*. Jika AUC bernilai 1, maka model dapat sempurna dalam membedakan kelas. Ilustrasi perhitungan AUC diperlihatkan oleh gambar 9 berikut:



Gambar 9. Area Under the Curve (AUC)

Classification report hanya menampilkan performa model dengan nilai *threshold* tertentu, sedangkan ROC-AUC mampu menunjukkan gambaran umum di seluruh nilai *threshold*.

b. Metrik Evaluasi pada Model Regresi

Dalam *machine learning*, model regresi digunakan untuk memprediksi data *continue*, seperti harga saham, suhu udara, jumlah penjualan dan lain-lain. Untuk menilai performa model regresi diperlukan metrik yang dapat menilai seberapa dekat hasil prediksi dengan nilai actual. Berbeda dengan metrik pada klasifikasi yang menekankan ketepatan kategori, metrik pada regresi berfokus pada ukuran kesalahan model. Beberapa metrik yang umum

digunakan dalam mengukur performa regresi antara lain: MAE, MSE, RMSE, dan Coefficient of Determination (R^2 Score).

Mean Absolute Error (MAE)

Mean Absolute Error merupakan salah satu metrik sederhana dalam menilai kinerja model regresi. MAE ditihung dari rata-rata selisih absolut antara nilai prediksi model dengan nilai actual. Secara matematis nilai MAE dirumuskan sebagai berikut:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{6}$$

Dimana:
n : jumlah data
y_i : nilai aktual
$\hat{y}_i$: nilai prediksi

Metrik MAE menunjukkan rata-rata besar kesalahan model dalam satuan yang sama dengan target. Misalnya : sebuah model prediksi harga rumah dalam juta rupiah memiliki nilai MAE sebesar 25, berarti rata-rata prediksi meleset 25 juta rupiah.

Mean Squared Error (MSE)

MSE mengukur rata-rata kuadrat seliish antara nilai prediksi model dengan nilai actual. Secara matematis nilai MSE dihitung dengan persamaan berikut:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{7}$$

Dimana:
n : jumlah data
y_i : nilai aktual
$\hat{y}_i$: nilai prediksi
Semakin kecil nilai MSE, maka semakin baik model regresi tersebut. Metrik MSE biasa digunakan dalam fungsi loss pada saat training model, hal ini dikarenakan sifat MSE yang memberikan penalty besar pada kesalahan prediksi ekstrem. MSE sangat sensitive terhadap outlier, sehingga hasil MSE pada data yang memiliki outlier yang banyak akan cenderung menyesatkan.

Root Mean Squared Error (RMSE)

RMSE merupakan nilai akar kuadrat dari MSE. Nilai RMSE di interpretasikan dengan satuan yang sama dengan target, Seperti pada MAE. Namun, RMSE tetap mempertahankan penalty terhadap kesalahan yang besar. Secara matematis, persamaan RMSE dapat dituliskan sebagai berikut:

$$RMSE = \sqrt{MSE} \tag{8}$$

R Squared(R^2)

R Squared atau Koefisien Determinasi merupakan metrik yang berfungsi untuk mengukur proporsi variansi (*variance proportion*) pada data target (nilai aktual) yang dapat dijelaskan oleh model prediksi. Secara matematis, fungsi $R\,Squared$ dapat dituliskan sebagai berikut:

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} \quad (9)$$

Dimana:
n : jumlah data
y_i : nilai aktual
$\hat{y}_i$: nilai prediksi
$\bar{y}$: rata-rata nilai Aktual
Pembilang $\sum(y_i - \hat{y}_i)^2$disebut Residual Sum of Squares (SSE)
Penyebut $\sum(y_i - \bar{y})^2$ disebut Total Sum of Square (SST)

Nilai R square tidak memberi informasi tentang besarnya error. Sebuah model dapat memiliki R square yang sama meskipun nilai MAE atau RMSEnya berbeda. Model dikatakan baik jika memiliki nilai R square mendekati 1. Jika nilai R square bernilai 1 berarti model dapat menjelaskan data dengan sempurna(prdiksi=aktual).

c. Metrik Evaluasi Pada *Object Detection*

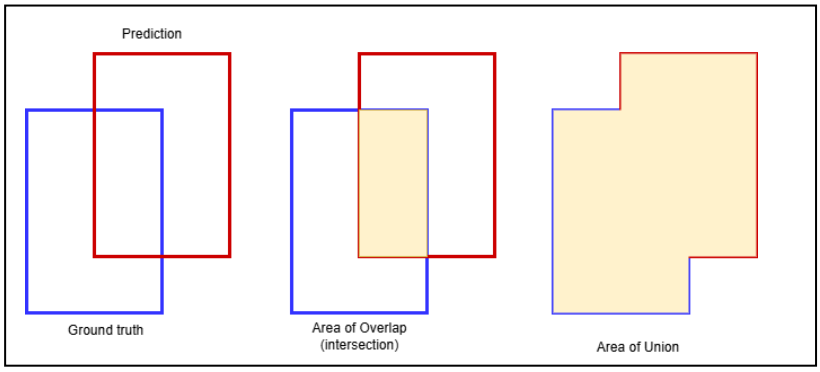
Dalam *object detection* memiliki tantangan evaluasi yang lebih kompleks dibandingkan dengan klasifikasi biasa. Hal ini karena model tidak hanya memprediksi kelas objek tetapi juga menentukan lokasi objek dalam citra melalui *bounding box*. Oleh karena itu object detection harus mampu menilai ketepatan klasifikasi sekaligus ketepatan lokalisasi. Beberapa metrik yang biasa digunakan dalam *object detection* antara lain IOU, AP, MAP, maupun FPS

Intersection Over Union (IOU)

IOU merupakan metrik dasar yang berfungsi untuk mengukur seberapa besar tumpang tindih antara *bounding box* prediksi dan *bounding box* pada *ground truth*. Secara matematis nilai IOU dapat dituliskan oleh persamaan berikut:

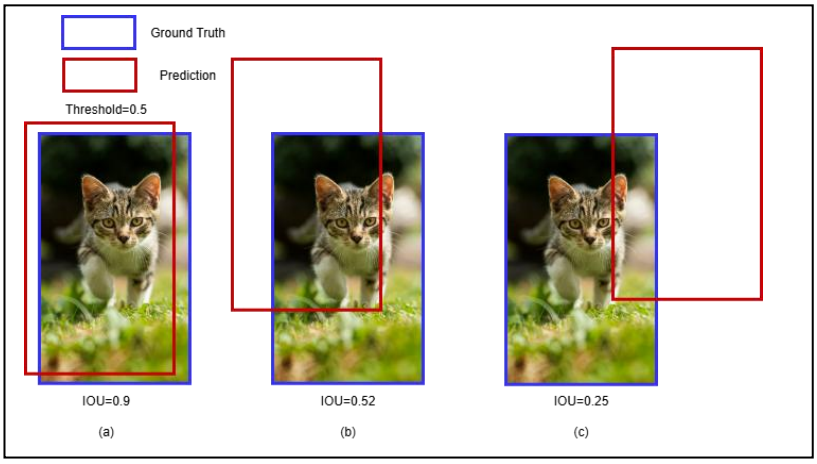
$$IOU = \frac{Area\ Of\ Overlap}{Area\ of\ Union} \quad (10)$$

Untuk memudahkan memahami *area of overlap* dan *area of union*, silakan lihat ilustrasi dari gambar 10 berikut:



Gambar 10. *Intersection Over Union (IOU)*

Untuk menentukan nilai *true positive* maupun *false positive* dari object yang di deteksi, biasanya akan menggunakan nilai threshold tertentu, dari nilai IOU model. Misalnya model deteksi objek kucing berikut menggunkan nilai *threshold* 0.5 untuk menentukan ketepatan objek yang di deteksi, maka gambaran nilai *true positive* dan *false positive* dapat di perlihatkan oleh gambar 11 berikut:

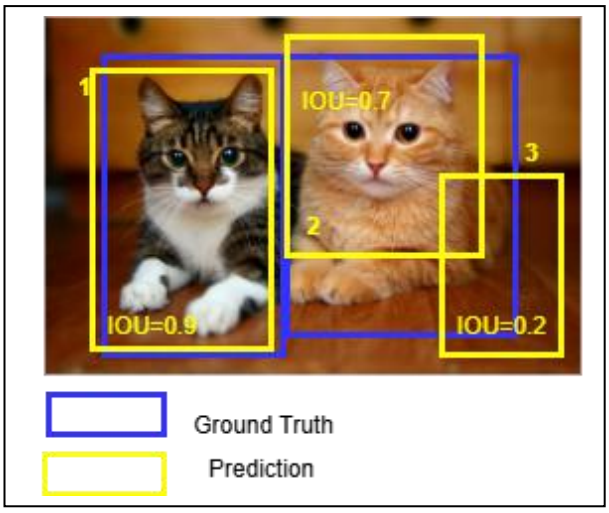


Gambar 11. IOU Pada Deteksi Kucing

Dengan menggunakan *threshold* 0.5 maka gambar 11. (a) & (b) yang memilki nilai IOU lebih dari *threshold* maka model berhasil mendeteksi objek (*true positive*), sedangkan pada gambar c, dengan IOU yang kurang dari *threshold* maka model gagal mendeteksi objek (*false positive*).

***Average Precision* (AP)**

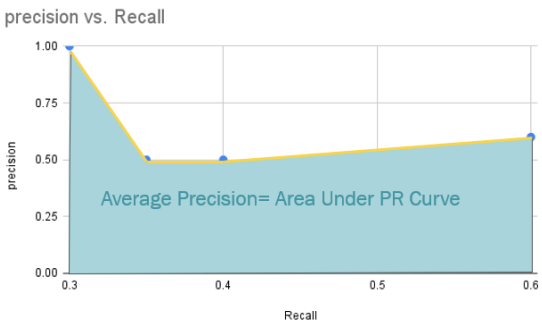
Precision Mengukur seberapa banyak deteksi model yang benar dibandingkan dengan semua deteksi yang dibuat. Sedangkankan *Recall* mengukur seberapa banyak objek-objek yang benar terdeteksi dibandingkan dengan total objek yang ada dalam gambar. Perhatikan contoh berikut ini:



Gambar 12. Precision & Recall Pada Object Detection

Pada gambar 12 tersebut jika kita menggunakan *threshold* 0.5, maka kita dapat menghitung nilai *precision* dan *recall* dari prediksi model. Prediksi 1 & 2, merupakan prediksi yang tepat (*true positive*), sedangkan prediksi 3 merupakan prediksi yang meleset (*false positive*). Sehingga nilai *precision*nya adalah $\frac{2}{3}$ sedangkan *recall*nya adalah $\frac{2}{2}$. Nilai *average precision* dihitung dengan luar area di bawah kurva *precision recall* untuk satu kelas object, seperti yang terlihat pada gambar 13 berikut:

Precision	Recall
1	0.3
0.5	0.35
0.6	0.6
0.5	0.4



Gambar 13. Average Precision

***Mean Average Precision* (MAP)**

Mean Average Precision berarti rata-rata dari *Average Precision* (AP) semua kelas. MAP menjadi standar utama dalam *benchmarking object detection*. Semakin tinggi MAP semakin baik performa sebuah model. Nilai AP dan MAP sangat bergantung pada *threshold* IOU yang digunakan. AP@0.5 berarti nilai *average precision* dengan nilai *threshold* IOU sebesar 0.5. untuk mendapatkan hasil yang general, kita bisa menggunakan berbagai nilai *threshold* IOU dalam menentukan AP dan MAP. Misalnya MAP dengan *threshold* IOU mulai dari 0.5 sampai dengan 0.95 dengan interval *threshold* 0.05 bisa kita notasikan dengan MAP@[0.50:0.95].

***Frame Per Second* (FPS)**

Selain ketepatan deteksi, kecepatan menjadi faktor yang penting dalam deteksi objek, terutama untuk aplikasi aplikasi berbasis *realtime* seperti cctv, drone atau *mobile otonom*. FPS menunjukkan jumlah *frame* yang bisa di proses per detik.

d. Metrik Evaluasi Pada LLM

Evaluasi *pada Large Language Model* (LLM) memiliki kompleksitas tersendiri karena tidak hanya menghasilkan nilai prediksi numerik tetapi juga teks yang panjang dan kontekstual. Ada beberapa metrik yang popular digunakan untuk mengukur performa LLM antara lain Bleu dan ROUGE

BLEU (*Bilingual Evaluation Understudy*)

Metrik ini banyak digunakan pada pada mesin penerjemah. Metrik ini diperkenalkan sejak 2002, oleh Papineni (Papineni 2002). Metrik ini dihitung dengan mengukur kesamaan N-gram (urutan kata) antara teks yang dihasilkan oleh model (*candidate translation*) dan teks referensi manusia (*reference translation*). Jika banyak $n-gram$ yang cocok, maka skor BLEU akan tinggi. BLEU biasanya dihitung untuk berbagai ukuran $n-gram$, biasanya dari unigram (1 kata) hingga $4-gram$ (frasa 4 kata). Secara matematis nilai BLEU dapat dihitung dengan persamaan berikut ini:

$$BLEU = BP.\exp\left(\sum_{n=1}^{N} w_n log p_n\right) \tag{11}$$

BP merupakan $brevity\ penalty$, yaitu nilai hukuman jika hasil terjemahan terlalu pendek dari referensi. n merupakan jumlah gram, p_n merupakan precision $n-gram$. Dan w_n merupakan nilai bobot positif.

$$BP\begin{cases}1 & if\ c > r \\ e^{\left(1-\frac{r}{c}\right)} & if c \leq r\end{cases} \tag{12}$$

c merupakan panjang *candidate translation* dan r merupakan panjang *corpus* dari reference.

ROUGE (*Recall-Oriented Understudy for Gisting Evaluation*)

Metrik ini popular digunakan dalam konteks peringkasan teks. ROUGE dikembangkan oleh Chin-Yew Lin pada tahun 2004 sebagai pendekatan berbasis perbandingan antara *candidate summary* (ringkasan yang dihasilkan oleh model) dengan (ringkasan acuan yang dibuat manusia).

ROUGE mengukur tumpang tindih (*overlap*) antara unit teks dalam kandidat ringkasan dengan referensi. unit yang dihitung dapat berupa $n-gram$, urutan kata (*sequence*), atau *subsequence*. Semakin tinggi tingkat kesamaan unit teks tersebut, semakin baik kualitas ringkasan atau hasil generasi yang diukur. Secara matematis $ROUGE-N$ dapat dituliskan sebagai berikut:

$$ROUGE-N = \frac{\sum_{S\in\{R\}}\sum_{gram_n\in S}\min\left(Count_c(gram_n), Count_s(gram_n)\right)}{\sum_{S\in\{R\}}\sum_{gram_n\in S} Count_s(gram_n)} \tag{13}$$

Keterangan:

- $gram_n$: potongan $n - gram$ dalam referensi
- $Count_c(gram_n)$: jumlah kemunculan $n - gram$ dalam kandidat
- $Count_s(gram_n)$: jumlah kemunculan $n - gram$ dalam referensi

e. Metrik Evaluasi Pada Clustering

Evaluasi klustering merupakan tahap penting dalam *unsupervised learning*, karena berbeda dengan klasifikasi yang memiliki label (*ground-truth)*, hasil klustering seringkali tidak memiliki label acuan. Oleh karena itu, diperlukan metrik evaluasi khusus untuk menilai kualitas hasil pengelompokan data. Secara umum, metrik evaluasi klustering dibagi menjadi dua kategori utama: *internal metrics* dan *external metrics*. *Internal metrics* berfokus untuk mengukur kepadatan (*compactness*) dan pemisahan (*separation*) antar *cluster*. Contoh *internal metrics* antara lain: *Silhouette Coefficient*, *Davies–Bouldin Index* (DBI), dan *Calinski–Harabasz Index* (CH). *Eksternal metrics* digunakan ketika tersedia label (*ground-truth*) sehingga hasil klustering dapat dibandingkan dengan pembagian kelas yang sebenarnya. Contoh *eksternal metric* antara lain: *Rand Index* (RI), *Purity*, *Normalized Mutual Information* (NMI) dan *Adjusted Rand Index* (ARI).

Silhouette Coefficient

Metrik *Silhouette Coefficient* pertama kali diperkenalkan oleh Rousseeuw (1987) untuk menilai kualitas pembentukan cluster berdasarkan kedekatan (cohesion) dan keterpisahan (separation) antar *cluster*. *Silhouette* memberikan gambaran apakah sebuah objek ditempatkan pada *cluster* yang benar atau seharusnya lebih dekat ke *cluster* lain. Secara matematis nilai *silhouette* untuk titik i dapat dihitung dengan persamaan berikut:

$$s(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}} \tag{13}$$

dengan $a(i)$ adalah jarak rata-rata titik i dengan cluster sendiri, dan $b(i)$ merupakan jarak rata-rata dengan cluster terdekat. Nilai *Silhouette Coefficient* berkisar antara -1 hingga 1. semakin mendekati 1, maka performa *clustering* semakin baik.

Davies–Bouldin Index (DBI)

Metrik ini diperkenalkan oleh Davies dan Bouldin pada tahun 1979 dengan tujuan memberikan ukuran kuantitatif terhadap keterpisahan (*separation*) antar cluster dan kemiripan intra cluster. Secara umum, DBI dirumuskan sebagai berikut:

$$DBI = \frac{1}{k}\sum_{i=1}^{k} \max_{j \neq i}\left(\frac{S_i + S_j}{M_{ij}}\right) \tag{14}$$

Dengan k adalah jumlah *cluster*, dan S_i merupakan rata-rata jarak antara setiap titik dalam cluster i dengan pusat *cluster* i. Nilai S_i dihitung dengan persamaan berikut:

$$S_i = \frac{1}{|C_i|}\sum_{x \in C_i} d(x, c_i) \tag{15}$$

M_{ij} merupakan jarak antar pusat *cluster* i dan cluster j, sedangkan rasio $\frac{S_i+S_j}{M_{ij}}$ menunjukkan seberapa besar kemiripan antara *cluster* i dan j. Semakin rendah nilai DBI, semakin baik kualitas *clustering*.

Adjusted Rand Index (ARI)

ARI merupakan pengembangan dari *Rand Index* (RI) yang diperkenalkan oleh William Rand (1971), dengan penyesuaian (*adjustment*) terhadap kemungkinan kesesuaian yang terjadi secara acak. Secara konseptual, Rand Index menilai kesesuaian hasil cluster dengan nilai acuan. Namun, kelemahan utama RI adalah tidak memperhitungkan bahwa sebagian kesesuaian dapat muncul karena kebetulan. ARI memperbaiki hal tersebut dengan menormalkan skor terhadap nilai ekspektasi acak, sehingga interpretasinya menjadi lebih adil.

Secara matematis nilai ARI dirumuskan sebagai berikut:

$$ARI = \frac{R - E}{\max(R) - E} \tag{16}$$

Dimana R Adalah nilai *Rand Index*, sedangkan E Adalah nilai ekspektasi *Rand Index* pada *cluster* acak. $Max\ (R)$ merupakan nilai maksimum yang dapat dicapai $Rand\ Index$, yaitu 1. Nilai $Rand\ Index$ dapat dihitung dengan persamaan:

$$R = \frac{a + b}{\binom{n}{2}} \tag{17}$$

Dimana:

a : jumlah pasangan elemen yang ditempatkan pada *cluster* yang sama baik dalam hasil *clustering* model maupun pada label acuan.

b : jumlah pasangan elemen yang ditempatkan pada cluster yang berbeda baik dalam hasil *clustering* model maupun pada label acuan.

n :jumlah total elemen dalam dataset

$\binom{n}{2}$:jumlah total pasangan elemen yang dapat dibentuk dari dataset (koefisien binomial)

Nilai *Adjusted Rand Index* berkisar antara -1 sampai dengan 1. Nilai 1 berarti hasil *clustering* identik dengan nilai acuan (hasil clustering baik).

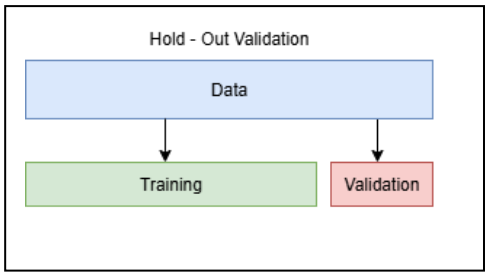
4. Mekanisme Evaluasi Secara *Offline*

Proses evaluasi *Machine Learning model* yang dilakukan secara daring di fase *prototyping* ini dapat dilakukan dengan beberapa pendekatan antara lain: *Hold – Out Validation, Cross Validation* maupun dengan *Bootstrap Resampling*.

a. Hold Out Validation

Hold out validation merupakan salah satu metode paling sederhana untuk mengevaluasi model *machine learning*. Konsep dari *holdout validation* adalah dengan membagi dataset menjadi 2 bagian, yakni *data training* dan *data testing*/data validasi seperti ilustrasi

Gambar 14. Data latih (*data training*) digunakan untuk membangun model, sedangkan *data testing*/validasi digunakan untuk menguji performa model.

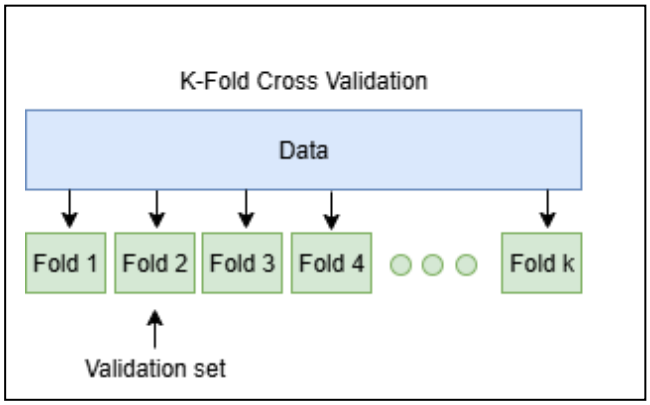


Gambar 14. *Hold-Out Validation*

Pembagian data dilakukan dengan proporsi, tergantung jumlah data yang tersedia. Misalnya 70:30, 80:20 atau yang lainnya. Untuk mendapatkan hasil yang lebih general, proses *hold-out validation* ini dapat dilakukan berulang kali dengan berbagai variasi ukuran *test set*.

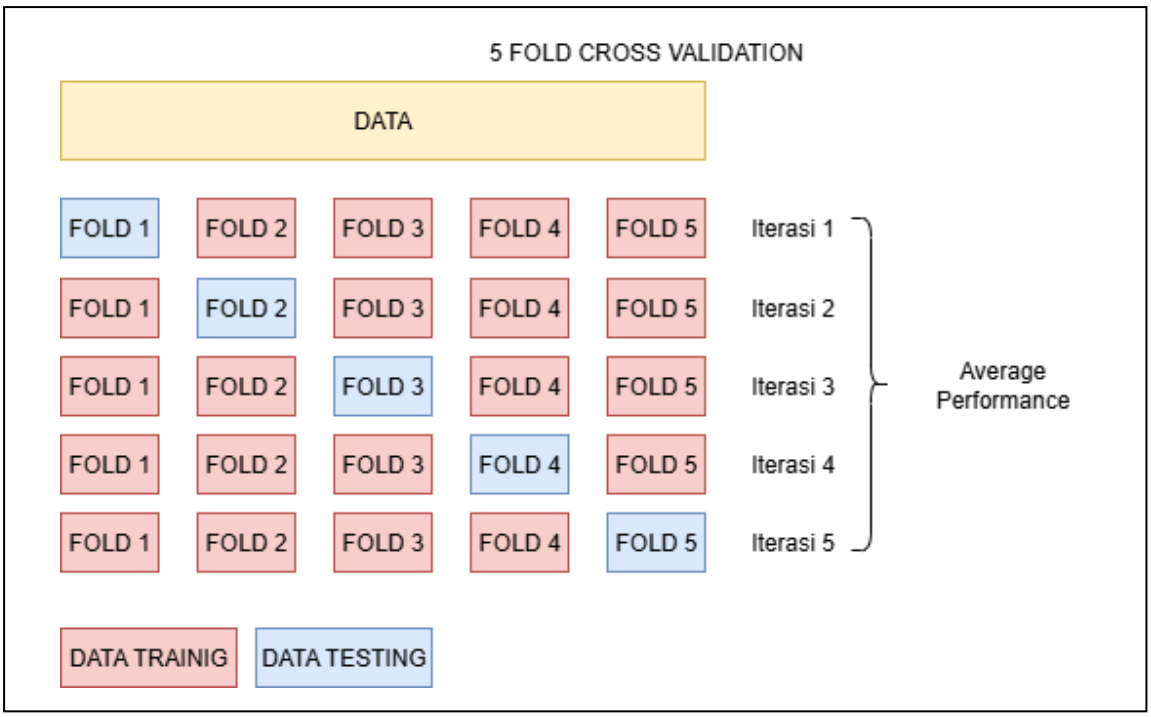
b. *K-Fold Cross Validation*

Cross-validation merupakan suatu metode statistik berbasis *resampling* yang banyak digunakan dalam machine learning untuk mengevaluasi kemampuan model pada sampel data yang terbatas. Teknik *K-Fold Cross Validation* dilakukan dengan membagi dataset ke dalam beberapa bagian (*fold*). K merupakan jumlah *fold* yang akan digunakan dalam membagi dataset. Ilustrasi pembagian *K-Fold Cross Validation* diperlihatkan oleh Gambar 15 berikut.



Gambar 15. Fold

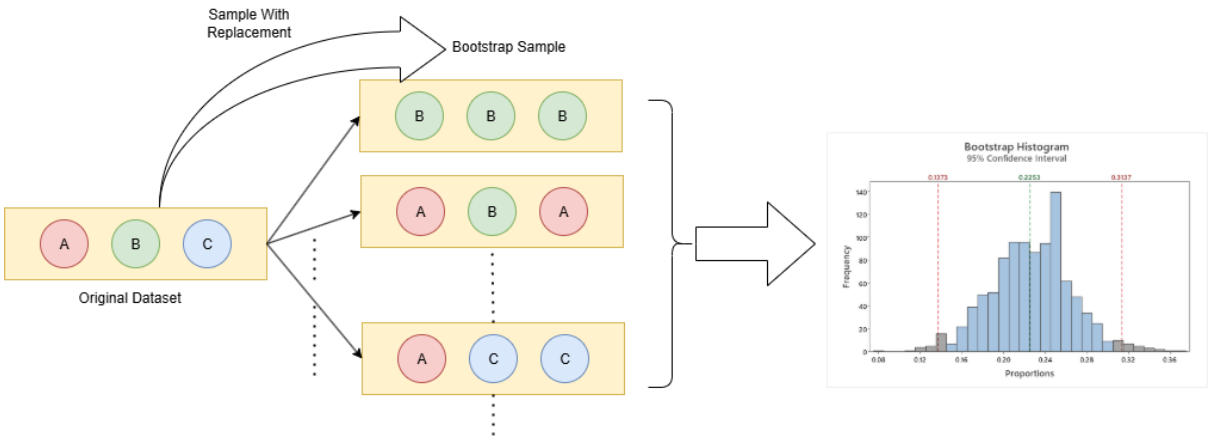
Setiap *fold* secara bergantian akan digunakan sebagai data uji, sementara *fold* yang lain digunakan untuk data latih. Model akan dilatih sebanyak k kali. Hasil evaluasi pada setiap iterasi digunakan untuk mengitung rata- rata kinerja model secara keseluruhan. Proses evaluasi menggunakan strategi *k-fold Cross validation* di perlihatkan oleh gambar 16 berikut



Gambar 16. 5-Fold Cross Validation

c. *Bootstrap Resampling*

Bootstrap dapat melakukan estimasi akurasi dari suatu model. *Bootstrap* merupakan metode *resampling* yang dikembangkan oleh Bradley Efron pada tahun 1979. Istilah "*bootstrap*" berasal dari pepatah Inggris "*pull oneself up by one's bootstraps*," yang berarti "membantu diri sendiri". Konsep dasar *bootstrap* adalah mengambil banyak sampel ulang (*resamples*) dari data asli dengan penggantian, lalu menghitung statistik dari setiap sampel ulang tersebut. Metode ini membangun estimasi statistik hanya dari data sampel yang ada, tanpa bergantung pada asumsi distribusi tertentu, seperti yang diperlihatkan oleh gambar 17 berikut:



Gambar 17. Bootstrap Resampling

Mekanisme evaluasi dengan bootstrap resempling dilakukan dengan melatih modeal dan mengevaluasi model pada setiap sampel *bootstrap*. Hasil evaluasi yang diperoleh

(misalnya accuracy, *precision*, *recall*, atau metrik yang lain) dikumpulkan sehingga membentuk distribusi empiris dari performa model.

5. **Mekanisme Evaluasi Secara Daring**

   Pada fase *deployment*, model *machine learning* tidak hanya dituntut untuk dapat melakukan prediksi, tetapi juga perlu dipantau dan dievaluasi secara berkelanjutan untuk menjamin konsistensi kinerjanya dalam lingkungan produksi. Evaluasi secara daring (*online evaluation*) merupakan mekanisme penting yang memungkinkan sebuah model dievaluasi berdasarkan performa aktual ketika berinteraksi dengan data dan pengguna secara langsung. *Realtime Monitoring* menggunakan dashboard analitik perlu dilakukan untuk mendeteksi penurunan performa secara dini. Adanya data drift[1] seringkali mengubah performa model ketika model sudah diterapkan (di deploy). Metode evaluasi lain yang dapat diterapkan di fase deployment Adalah dengan A/B testing, yaitu dengan membagi user(pengguna) ke dalam dua kelompok. Kelompok A menggunan model lama (Baseline model), dan kelompok B menggunakan model baru yang sedang dievaluasi. Perbandingan hasil dari kedua kelompok (misalnya tingkat kepuasan pengguna, konversi, atau efisiensi) memberikan data empiris untuk memutuskan apakah model baru layak diadopsi secara penuh.

6. **Referensi**

[1] A. Zheng, Evaluating Machine Learning Models, 1st ed. O'Reilly Media, 2015.

[2] L. A. Yates, Z. Aandahl, S. A. Richards, and B. W. Brook, "Cross validation for model selection: A review with examples from ecology," Ecol Monogr, vol. 93, no. 1, Feb. 2023, doi: 10.1002/ecm.1557.

[3] B. Efron, "Bootstrap Methods: Another Look at the Jackknife," 1979.

[4] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: a Method for Automatic Evaluation of Machine Translation."

7. **Credit**

   Icon:

   - **smashingstocks from www.flaticon.com**
   - **pixel perfect from www.flaticon.com**

[1] fenomena perubahan distribusi statistik pada data input (fitur) yang digunakan oleh model *machine learning* setelah model tersebut di-*deploy* ke lingkungan produksi