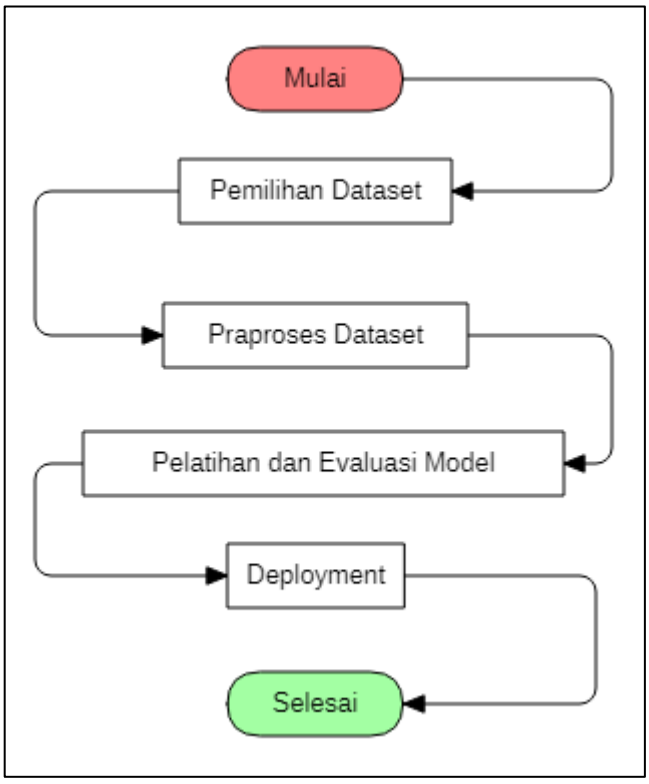
BAB III

METODOLOGI PENELITIAN

3.1 Tahapan Penelitian

Penelitian ini menggunakan metode rekayasa perangkat lunak (*software engineering*) dengan pendekatan eksperimen kuantitatif. Proses penelitian melibatkan pengembangan sistem berbasis *Machine Learning*, yang kemudian diuji dan dievaluasi performanya menggunakan metrik kuantitatif untuk menilai efektivitas model dalam melakukan klasifikasi.

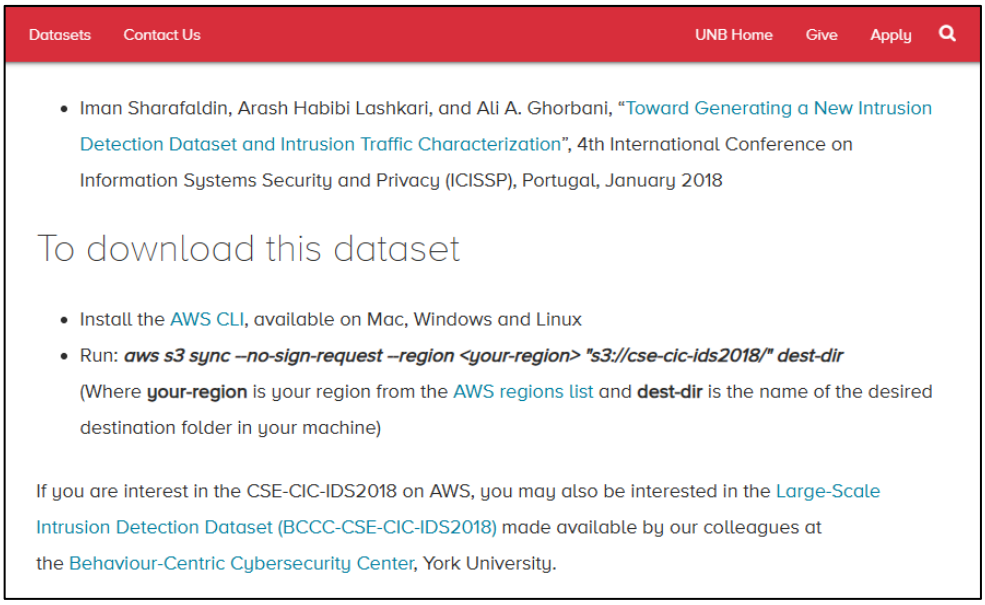


Gambar III.1 Tahapan Pengembangan Aplikasi

Berdasarkan Gambar III.1, tahapan penelitian secara umum dimulai dari tahap pemilihan *dataset*, dilanjutkan dengan praproses data, pelatihan dan evaluasi model, hingga tahap *deployment* atau penerapan model ke dalam sebuah aplikasi REST API yang dikembangkan dalam bentuk PCAP *Analyzer*. Aplikasi ini dirancang untuk menerima dan menganalisis *file* PCAP yang merupakan *file* standar yang umum digunakan dalam proses forensik digital. Dengan demikian, pemanfaatan PCAP *Analyzer* dapat mempermudah pengguna yang terbiasa melakukan investigasi forensik digital dalam mengakses dan menginterpretasikan hasil klasifikasi serangan.

3.1.1 Pemilihan *Dataset*

Penelitian ini memanfaatkan *dataset* CSE-CIC-IDS2018, atau dikenal sebagai CICIDS2018, yang berasal dari *University of New Brunswick*. *Dataset* CICIDS2018 mencakup tujuh skenario serangan berbeda, yaitu *Brute-force*, *Heartbleed*, *Botnet*, *DoS*, *DDoS*, serangan web, dan infiltrasi jaringan internal. Infrastruktur yang digunakan untuk simulasi serangan melibatkan 50 komputer penyerang, sementara infrastruktur yang diserang terdiri dari lima departemen dengan total 420 komputer dan 30 *server*. *Dataset* ini merekam lalu lintas jaringan dan *log* sistem dari setiap komputer, serta membuat 80 variabel atau fitur dari lalu lintas yang diekstrak menggunakan *CICFlowMeter-V3*.



Gambar III.2 Situs yang Menyediakan *Dataset* CICIDS2018

Sumber *dataset* CICIDS2018 dapat diakses melalui situs resmi *Canadian Institute for Cybersecurity* (CIC) pada halaman *Research Dataset*. Proses pengunduhan *dataset* mengikuti panduan yang ditampilkan pada Gambar III.2, di mana penggunaan AWS CLI diperlukan untuk mengakses *file* yang tersedia di repositori penyimpanan daring tersebut.

Pemilihan *dataset* CICIDS2018 dalam penelitian ini didasarkan pada ketersediaan alat bantu konversi *file* PCAP ke *file* CSV, yakni *CICFlowMeter-v3*, yang dapat diunduh melalui repositori *GitHub* bernama TCPDUMP_and_*CICFlowMeter* pada direktori "*CICFlowMeters/CICFlowMeter-3.0*". Keberadaan alat ini mempermudah proses ekstraksi fitur atau variabel dari *file* PCAP sehingga mempercepat tahapan praproses data untuk pelatihan model *Machine Learning*.

3.1.2 Praproses Dataset

Tahap berikutnya adalah praproses data. Data mentah yang diperoleh memerlukan pemrosesan karena masih banyak fitur yang tidak relevan. Praproses data dilakukan untuk membersihkan data dan menstandardisasi nilai-nilainya. Langkah-langkah praproses ini penting agar pelatihan model menjadi lebih baik, serta untuk menghindari *bias* atau *variance* yang berlebihan yang berpotensi memengaruhi prediksi (*output*) model. Dalam proses ini, beberapa langkah utama harus dilakukan:

1. Menghilangkan nilai *NaN* dan *Infinity*

Langkah ini bertujuan untuk menghilangkan data yang tidak *valid*, *NaN* atau nilai hilang (*Not a Number*), atau memiliki nilai tak terhingga (*Infinity*). Keberadaan nilai-nilai ini dapat mengganggu perhitungan dan menyebabkan *error* dalam proses pelatihan model.

2. Transformasi logaritmik

Transformasi ini diterapkan pada fitur-fitur tertentu untuk mengurangi kemiringan (*skewness*) distribusi data dan menstabilkan *variance*. Hal ini sangat berguna untuk variabel dengan rentang nilai yang sangat lebar, sehingga data menjadi lebih sesuai untuk algoritma *Machine Learning*.

3. Transformasi variabel "*port*"

Transformasi pada variabel "*port*" ini dilakukan dengan mengubah datanya agar kardinalitas dari variabel tersebut berkurang, dengan cara mengubah nilai

minoritas menjadi nilai lain, setelah kardinalitas variabel berkurang, maka variabel "*port*" dapat diubah dengan *one-hot encoding*.

3.1.3 Pelatihan dan Evaluasi

Membuat model-model dari beberapa algoritma *Machine Learning* untuk melakukan klasifikasi serangan *malware* atau yang lainnya dilakukan setelah data selesai dipraproses. Pelatihan model ini dilakukan menggunakan bahasa program *Python* dan kumpulan *library* yang umum digunakan untuk proses *Machine Learning* seperti *Scikit-learn, Numpy, Matplotlib, Pandas* dan lainnya. Dari semua model yang dibuat akan dilakukan evaluasi.

Random Forest, *AdaBoost*, dan *Gradient Boosting* adalah tiga algoritma *ensemble Machine Learning* yang dipilih dalam penelitian ini karena banyak penelitian yang menunjukan performa yang tinggi ketika melakukan tugas klasifikasi terhadap banyak *dataset* NIDS, terutama algoritma *Random Forest*. *Random Forest* beroperasi berdasarkan prinsip *bagging* (*bootstrap aggregating*), di mana kumpulan *Decision Tree* dibangun secara independen dari subset data pelatihan yang diambil secara acak dan subset fitur yang dipilih secara acak, pemilihan subset dari *dataset* ini disebut juga sebagai teknik *bootstraping*. Prediksi akhir kemudian ditentukan melalui *voting* mayoritas, strategi yang efektif untuk mengklasifikasikan data multi kelas dengan mengurangi risiko *overfitting*.

AdaBoost (*Adaptive Boosting*) menggunakan pendekatan *boosting* yang bersifat sekuensial, secara iteratif membangun model dasar, seringkali berupa *Decision Tree* sederhana, dan memberikan bobot lebih pada sampel yang sulit diklasifikasikan dari iterasi sebelumnya. Dengan demikian, *AdaBoost* secara

adaptif memfokuskan pembelajaran pada area yang bermasalah, meningkatkan kinerja model secara bertahap.

Gradient Boosting merupakan algoritma *boosting* sekuensial dan berfokus untuk meminimalkan *loss function* dari model secara keseluruhan. Setiap model baru yang ditambahkan dilatih untuk memprediksi residu atau kesalahan dari model sebelumnya, kemudian kontribusinya digabungkan secara progresif untuk mengurangi *error*. Fleksibilitas ini membuat *Gradient Boosting* sangat kuat dan serbaguna dalam menangani berbagai masalah klasifikasi *multiclass*.

Random Forest, *AdaBoost*, dan *Gradient Boosting* dapat digunakan untuk melatih model secara mudah dengan bantuan *library Scikit-learn* di *Python*, *Scikit-learn* juga menyediakan fungsi-fungsi lain yang bisa digunakan untuk membantu dalam proses praproses data atau evaluasi model-model yang telah dilatih.

Evaluasi dilakukan untuk mengetahui apakah model yang dilatih siap untuk di-*deploy* ke sebuah aplikasi, jika model yang dilatih tidak mendapatkan performa yang baik, maka perlu dilakukan praproses data dengan tahapan yang berbeda, ini dikarenakan performa model sangat berkaitan erat dengan data yang diberikan. Evaluasi model dilakukan sebanyak dua kali, evaluasi pertama bertujuan untuk menentukan apakah performa model bagus, jika tidak maka tahapan praproses data perlu diubah. Tahapan yang kedua untuk menilai performa sesungguhnya dari model dan yang menjadi hasil dari laporan penelitian.

3.1.4 *Deployment*

Model yang dilatih dan mendapatkan performa optimal akan diimplementasikan ke dalam sebuah aplikasi web REST API yang ditulis dengan

bahasa program *Python*. Aplikasi ini dirancang untuk menerima *file* PCAP, mengubahnya menjadi *file* CSV yang sesuai untuk proses analisis model. Selanjutnya, hasil *file* CSV akan digunakan oleh program prediktor khusus yang bertanggung jawab untuk melakukan prediksi.