BAB V PENUTUP

5.1 Kesimpulan

Penggunaan AI-driven testing dalam pengujian perangkat lunak memberikan solusi yang lebih efisien dan andal dibandingkan dengan pengujian manual. Setelah melakukan serangkaian tahapan pada metodologi penelitian, didapatkan beberapa Kesimpulan sebagai berikut.

a. Pendekatan prediksi *software defects* berbasis *machine learning* mampu membantu *tester* dalam memprediksi *software defects* sejak tahap awal dalam Software Development Life Cycle (SDLC). Dengan menggunakan *software metrics* sebagai dataset dapat dibangun sebuah model prediksi yang berhasil mengklasifikasikan *package* yang berpotensi rentan terhadap *defects* dan yang *non-defects*. Model prediksi dibuat dengan menggunakan *tree-based ensemble* antara lain AdaBoost, Random Forest, Extra Trees, Gradient Boosting, Histogram-based Gradient Boosting, XGBoost, dan CatBoost. Model tersebut dioptimasi untuk meningkatkan nilai performa dengan menggunakan metodde *stacking ensemble.* Hasil evaluasi menunjukkan bahwa model Stacking memiliki performa terbaik dibandingkan model individual lainnya, dengan nilai ROC-AUC mencapai 0.8575. Hasil tersebut berhasil memecahkan rumusan masalah yakni meningkatkan performa dibandingkan dengan penelitian sebelumnya (0.7988).
b. Dengan melakukan *deployment* model pada *Robotic Process Automation,* memudahkan pemanfaatan model prediksi tersebut. Dengan keunggulan framework UiPath yang mampu menjalankan *powershell* pada local, memudahkan penggunaan model prediksi hanya dengan menginputkan link repository nya saja dan tidak perlu mengekstrak terlebih dahulu project nya untuk mendapatkan data *software metrics.* Proses ini sangat efisien karena hanya memakan waktu 2-6 menit bergantung pada kompleksitas project yang diprediksi.

5.2 Saran

Berdasarkan hasil skripsi, terdapat beberapa saran atau rekomendasi untuk pengembangan prediksi *software defects* lebih lanjut, antara lain.

a. Melalui tahapan eksplorasi pemilihan *software metrics,* didapatkan pemahaman karakteristik dan keterhubungan setiap atribut *software metrics* terhadap potensi *defects*. Hal ini diharapkan bisa menjadi wawasan untuk melakukan penelitian atribut *software metrics* yang memiliki karakteristik atribut yang mendekati dan merepresentasikan hubungan antara *software metrics* dengan potensi *defects* pada bahasa pemrograman lain, baik pada bahasa pemrograman berorientasi object (OOP) ataupun lainnya.
b. Sebagai pencegahan dataset yang tidak seimbang (*imbalanced* dan data yang terlalu kecil, disarankan untuk memperbanyak *dataset* secara organik agar dapat meningkatkan nilai keterbacaan untuk memprediksi kelas yang *bugs* (*True Positive).* meningkatkan generalisasi model dan potensi peningkatan performa, termasuk akurasi prediksi. Dengan data yang lebih representatif dari berbagai proyek perangkat lunak, model memiliki peluang lebih tinggi untuk mendeteksi pola cacat yang kompleks dan meningkatkan nilai keterbacaan untuk memprediksi kelas yang *bugs* (*True Positive).*
c. Sebagai Langkah peningkatan kebermanfaatan model prediksi, penelitian selanjutnya dapat mengembangkan model prediksi *software defects* dengan menggunakan pendekatan *explainable AI.* Hal ini memungkinkan *developer* maupun *software tester* mendapatkan alasan rasional yang membuat suatu modul diprediksi *defects.* Pendekatan ini memungkinkan pengembang memahami faktor-faktor utama yang memengaruhi hasil prediksi, sehingga tidak hanya meningkatkan kepercayaan terhadap model, tetapi juga membantu dalam pengambilan keputusan selama proses pengembangan perangkat lunak.