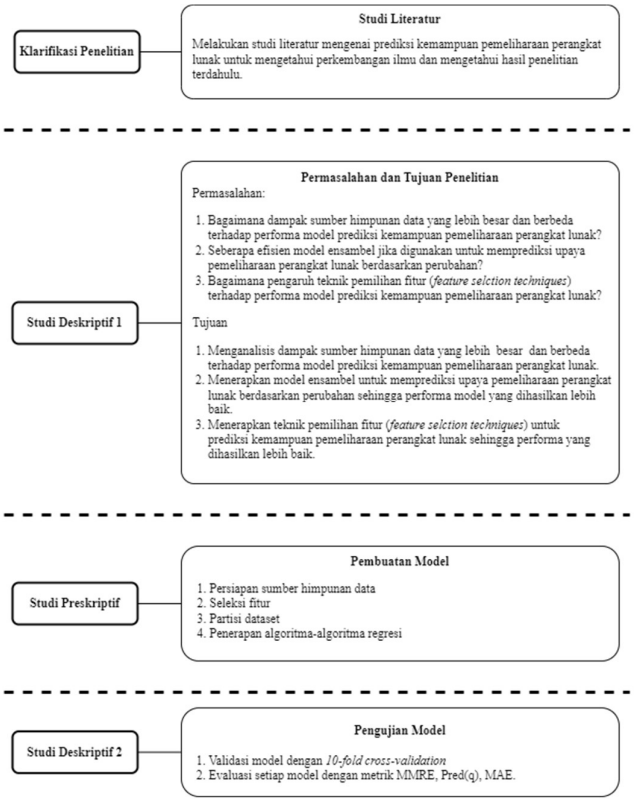
BAB III
METODE PENELITIAN

3.1 Desain Penelitian

Desain penelitian yang akan digunakan pada penelitian ini adalah dengan menggunakan metode *Desain Research Methodology* (DRM). DRM terdiri dari empat tahap yaitu Klarifikasi Penelitian atau *Research Clarification* (RC), Studi Deskriptif I atau *Descriptive Study I* (DS-I), Studi Preskriptif atau *Perscriptive Study* (PS), dan Studi Deskriptif II *Descriptive Study II* (DS-II). Dalam DRM jika penelitian yang dilakukan melalui tahap RC hingga DS-II tanpa iterasi, maka tipe penelitian tersebut termasuk ke dalam Pengembangan Pendukung (*Development of Support*) (Blessing & Chalkrabarti, 2009). Gambar 3.1 menunjukkan skema penelitian yang akan di lakukan dengan berdasarkan kerangka kerja DRM.



Gambar 3.1 Skema Penelitian

Berikut merupakan penjelasan dari Skema Penelitian pada Gambar 3.1:

1. Klarifikasi Penelitian

   Pada tahap ini, studi literatur dilakukan untuk mengetahui perkembangan ilmu, mengetahui hasil penelitian terdahulu, memperjelas dan memfokuskan tujuan penelitian. Studi literatur menggunakan penelitian-penelitian yang relevan dan buku-buku untuk memperdalam pengetahuan tentang bidang yang diteliti.

2. Studi Deskriptif I

   Pada tahap ini, permasalahan dan tujuan penelitian berdasarkan studi literatur di definisikan. Permasalahan merupakan pertanyaan-pertanyaan yang akan dipecahkan oleh penelitian yang dibuat berdasarkan latar belakang. Terdapat beberapa pertanyaan yang menunjukkan fokus permasalahan penelitian. Tujuan merupakan hasil yang diharapkan dan ingin dicapai pada penelitian yang akan dilakukan.

3. Studi Preskriptif

   Pada tahap ini, langkah-langkah pembuatan model prediksi tingkat kemampuan pemeliharaan perangkat lunak diimplementasikan. Langkah-langkah yang dilakukan mengacu beberapa penelitian terdahulu yang kemudian digabungkan untuk meningkatkan performa model yang dibuat.

4. Studi Deskriptif 2

   Pada tahap ini, metode-metode pengujian model regresi digunakan untuk mengevaluasi performa model prediksi tingkat kemampuan pemeliharaan perangkat lunak. Beberapa metrik yang digunakan yaitu MMRE, Pred, dan MAE. Kemudian, teknik validasi digunakan untuk mengevaluasi performa model yang telah dilatih dengan menggunakan *10-fold cross-validation*.

3.2 Sumber Himpunan Data

Sumber himpunan data yang bersifat publik banyak digunakan oleh para peneliti dengan alasan mudah di replikasi dan memudahkan peneliti lain untuk membandingkan dengan penelitian mereka di masa yang akan datang (Kumar & Rath, 2017). Sumber himpunan data yang digunakan dalam penelitian ini adalah *dataset* publik yang tersedia di repositori *Zenodo* yang dibuat oleh Hadeel Alsolai (H. Alsolai, 2020). *Dataset software maintainability prediction* ini ditulis dalam

bahasa pemrograman Java dan menyediakan data di tingkat kelas (*class level*) yang terdiri dari lima *dataset* sistem perangkat lunak yang terdaftar. Adapun deskripsi mengenai kelima *dataset* sistem perangkat lunak tersebut dijelaskan dalam Tabel 3.1.

Tabel 3.1

Deskripsi Dataset Sistem Perangkat Lunak pada *Software Maintainability Prediction Dataset*

No.	Nama Dataset	Deskripsi
1	Eclipse JDT Core	JDT Core adalah infrastruktur Java dari Java IDE.
2	Eclipse PDE UI	Komponen UI PDE menyediakan seperangkat alat yang komprehensif untuk membuat, mengembangkan, menguji, men-debug, dan men-deploy plug-in Eclipse, fragmen, fitur pembaharuan situs dan produk RCP.
3	Equinox Framework	Equinox Framework adalah implementasi dari spesifikasi kerangka inti OSGi, satu set bundle yang mengimplementasikan berbagai layanan OSGi opsional dan infrastruktur lainnya untuk menjalankan sistem berbasis OSGi.
4	Lucene	Lucene adalah sebuah library Java yang menyediakan fitur pengindeksan dan pencarian yang kuat, pemeriksaan ejaan, dan kemampuan analisis terbaik.
5	Mylyn	Mylyn adalah kerangka kerja manajemen siklus hidup tugas dan aplikasi atau *application lifecycle management* (ALM) untuk Eclipse.

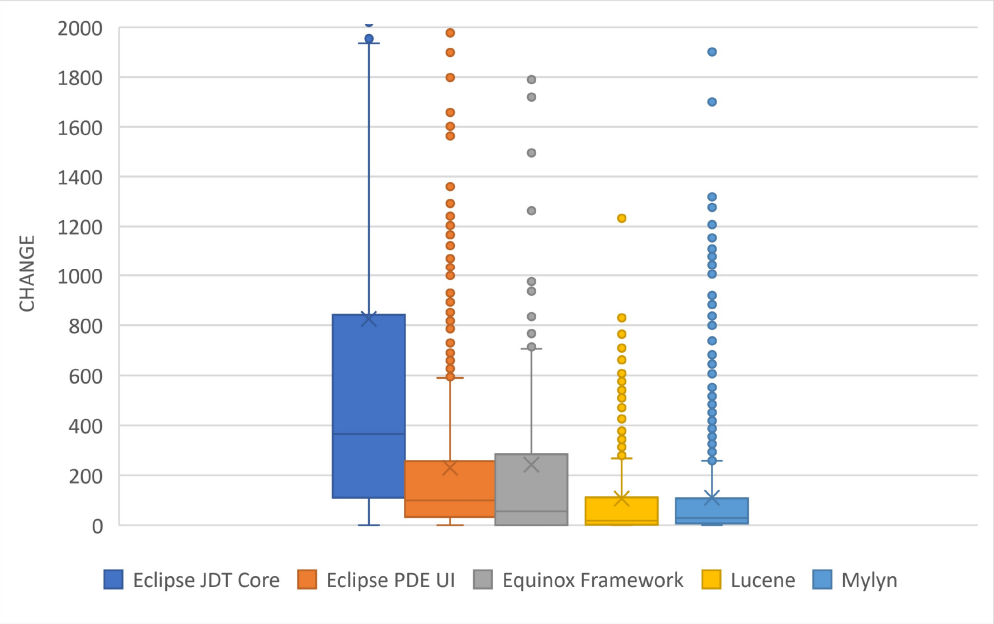
Dataset ini telah dibuat dalam sebuah penelitian disertasi dan telah melewati beberapa proses pra-pemrosesan data seperti reduksi data (*data reduction*), integrasi data (*data integration*), pembersihan data (*data cleaning*), dan transformasi data (*data transformation*). Kemudian, *dataset* ini memiliki total sebanyak 4285 kelas/baris dan memiliki 18 metrik/atribut. Sebanyak 17 metrik merupakan variabel independen dan 1 metrik merupakan variabel dependen. Variabel dependen yaitu metrik CHANGE (perubahan) dapat diidentifikasi dengan 17 metrik variabel independen tersebut. Metrik CHANGE dijadikan acuan seberapa upaya yang dikeluarkan dalam pemeliharaan. Nilai metrik yang lebih tinggi mengacu pada upaya pemeliharaan yang lebih tinggi, yang mengindikasikan rendahnya tingkat kemampuan pemeliharaan. Sebaliknya, nilai metrik yang lebih

rendah mengacu pada upaya pemeliharaan yang lebih rendah, yang mengindikasikan tingginya tingkat kemampuan pemeliharaan (H. A. Alsolai, 2020). Untuk mengetahui distribusi data pada variabel dependen, statistik deskriptif pada setiap *dataset* untuk metrik CHANGE ditunjukkan oleh Tabel 3.2 dan *boxplot* dari metrik CHANGE ditunjukkan oleh Gambar 3.2.

Tabel 3.2

Statistik Deskriptif Metrik CHANGE

Nama Dataset	Jumlah Kelas	Mean	Std	Min	Q1	Q2	Q3	Max
Eclipse JDT Core	695	825.25	2013.10	0	110.00	366	839.00	31245
Eclipse PDE UI	1209	230.41	424.56	0	33.00	100	256.00	6824
Equinox Framework	276	242.59	520.87	0	0.00	56	283.50	5684
Lucene	539	106.97	247.50	0	3.75	19	112.00	3089
Mylyn	1537	110.80	316.04	0	8.00	29	108.00	9000



Gambar 3.2 *Boxplot* Metrik CHANGE

Diagram *boxplot* menggambarkan nilai kuartil pertama (Q1), kuartil kedua (Q2) atau median, kuartil ketiga (Q3), batas bawah, batas atas, dan nilai *whiskers*. Diagram *boxplot* yang didapat menunjukkan *boxplot* menceng ke arah kanan, yang berarti bahwa terdapat beberapa nilai *outlier* yang lebih tinggi dari nilai tertinggi

yang tercantum di dalam setiap dataset. Nilai *outlier* adalah nilai yang terpisah jauh dari kebanyakan nilai yang ada di dalam himpunan data.

Berdasarkan tabel dan diagram tersebut dapat diketahui bahwa nilai median dari dataset Eclipse JDT Core lebih tinggi dibanding dataset lainnya, yang berarti dataset Eclipse JDT Core memiliki upaya pemeliharaan yang lebih tinggi dan mengindikasikan rendahnya tingkat kemampuan pemeliharaan. Dataset Lucene memiliki nilai median yang lebih rendah dibanding dataset lainnya, yang berarti dataset Lucene memiliki upaya pemeliharaan yang lebih rendah dan mengindikasikan tingginya tingkat kemampuan pemeliharaan. Kemudian, deskripsi dari metrik-metrik variabel independen yang ada pada sumber himpunan data yang digunakan dijelaskan pada Tabel 3.3.

Tabel 3.3

Metrik-Metrik pada *Software Maintainability Prediction* Dataset

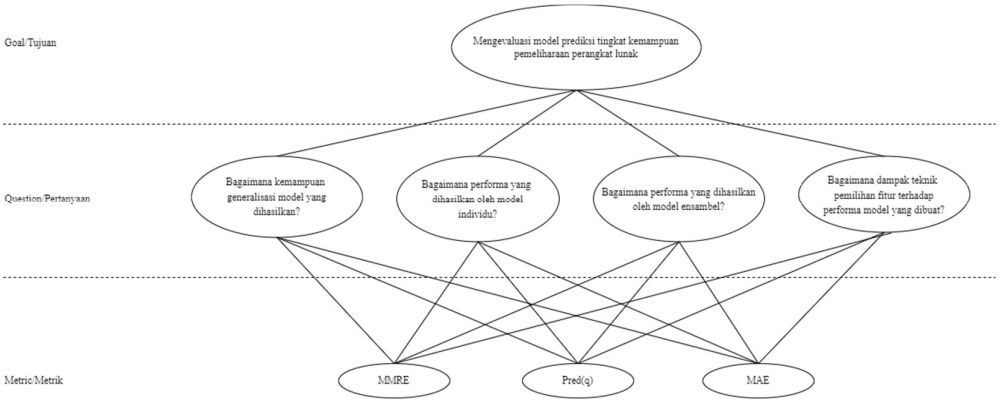
No.	Nama Metrik	Deskripsi
CK Metrik (Chidamber & Kemerer)		
1	*Lack of Cohesion in Methods* (LCOM)	LCOM adalah salah satu metrik yang digunakan untuk mengukur tingkat kohesi dari sebuah kelas atau objek. Kohesi adalah ukuran seberapa integrasi atau terkait satu sama lainnya sebuah kelas atau objek.
2	*Number of Children* (NOC)	NOC adalah salah satu metrik yang digunakan untuk mengukur jumlah kelas anak (*child class*) yang diwarisi oleh sebuah kelas. NOC dihitung dengan menghitung jumlah kelas anak yang mewarisi sebuah kelas, termasuk kelas anak dari kelas anak yang diwarisi.
3	*Depth of Inheritance Tree* (DIT)	DIT adalah salah satu metrik yang digunakan untuk mengukur tingkat kedalaman dari seluruh hierarki kelas dalam suatu program. DIT dihitung dengan menghitung jumlah kelas di antara kelas paling bawah dalam hierarki dan kelas paling atas, termasuk kelas paling atas itu sendiri.
4	*Coupling Between Objects* (CBO)	CBO adalah salah satu metrik yang digunakan untuk mengukur tingkat keterkaitan antar kelas atau objek dalam suatu program. CBO dihitung dengan menghitung jumlah kelas atau objek yang terkait dengan sebuah kelas atau objek, termasuk kelas atau objek yang digunakan oleh metode dari kelas atau objek tersebut.
5	*Response for Class* (RFC)	RFC adalah salah satu metrik yang digunakan untuk mengukur tingkat kompleksitas dari sebuah kelas atau objek. RFC dihitung

		dengan menghitung jumlah metode yang dipanggil oleh sebuah kelas atau objek, termasuk metode yang dipanggil oleh metode lain dari kelas atau objek tersebut.
6	Weighted Method Count (WMC)	WMC adalah salah satu metrik yang digunakan untuk mengukur tingkat kompleksitas dari sebuah kelas atau objek. WMC dihitung dengan menghitung jumlah metode yang ada dalam sebuah kelas atau objek, dan memberi bobot terhadap setiap metode berdasarkan tingkat kompleksitasnya.
OO Metrik		
7	FanIn	FanIn adalah salah satu metrik yang digunakan untuk mengukur jumlah kelas lain yang merujuk ke sebuah kelas. Metrik ini dihitung dengan menghitung jumlah kelas lain yang mengacu ke sebuah kelas, termasuk kelas yang mengacu ke kelas tersebut secara tidak langsung.
8	FanOut	FanOut adalah salah satu metrik yang digunakan untuk mengukur jumlah kelas lain yang ditunjuk oleh sebuah kelas. Metrik ini dihitung dengan menghitung jumlah kelas lain yang ditunjuk oleh sebuah kelas, termasuk kelas yang ditunjuk secara tidak langsung.
9	*Number of Attributes* (NOA)	NOA adalah salah satu metrik yang digunakan untuk mengukur jumlah atribut yang ada pada sebuah kelas atau objek. NOA dihitung dengan menghitung jumlah atribut yang didefinisikan dalam sebuah kelas atau objek, termasuk atribut yang didefinisikan dengan akses *modifier* "*private*" atau "*public*".
10	*Number of Attributes Inherited* (NOAI)	NOAI adalah salah satu metrik yang digunakan untuk mengukur jumlah atribut yang diwarisi oleh sebuah kelas. NOAI dihitung dengan menghitung jumlah atribut yang diwarisi oleh sebuah kelas, termasuk atribut yang diwarisi dari kelas anak yang diwarisi.
11	*Lines of Code* (LOC)	LOC adalah salah satu metrik yang digunakan untuk mengukur jumlah baris kode dalam suatu kelas. LOC dihitung dengan menghitung jumlah baris kode yang ada dalam suatu program, termasuk baris kode komentar dan baris kosong.
12	*Number of Methods* (NOM)	NOM adalah salah satu metrik yang digunakan untuk mengukur jumlah metode yang ada dalam sebuah kelas atau objek. NOM dihitung dengan menghitung jumlah metode yang didefinisikan dalam sebuah kelas atau objek, termasuk metode yang didefinisikan dengan akses *modifier* "*private*" atau "*public*".

13	*Number of Method Inherited* (NOMI)	NOMI adalah salah satu metrik yang digunakan untuk mengukur jumlah metode yang diwarisi oleh sebuah kelas. NOMI dihitung dengan menghitung jumlah metode yang diwarisi oleh sebuah kelas, termasuk metode yang diwarisi dari kelas anak yang diwarisi.
14	*Number of Private Attibutes* (NOPRA)	NOPRA adalah salah satu metrik yang digunakan untuk mengukur jumlah atribut privat yang ada dalam sebuah kelas atau objek. NOPRA dihitung dengan menghitung jumlah atribut yang didefinisikan dengan akses *modifier* "*private*" dalam sebuah kelas atau objek.
15	*Number of Private Methods* (NOPRM)	NOPRM adalah salah satu metrik yang digunakan untuk mengukur jumlah metode privat yang ada dalam sebuah kelas atau objek. NOPRM dihitung dengan menghitung jumlah metode yang didefinisikan dengan akses *modifier* "*private*" dalam sebuah kelas atau objek.
16	*Number of Public Attributes* (NOPA)	NOPA adalah salah satu metrik yang digunakan untuk mengukur jumlah atribut publik yang ada dalam sebuah kelas atau objek. NOPA dihitung dengan menghitung jumlah atribut yang didefinisikan dengan akses *modifier* "*public*" dalam sebuah kelas atau objek.
17	*Number of Public Methods* (NOPM)	NOPM adalah salah satu metrik yang digunakan untuk mengukur jumlah metode publik yang ada dalam sebuah kelas atau objek. NOPM dihitung dengan menghitung jumlah metode yang didefinisikan dengan akses *modifier* "*public*" dalam sebuah kelas atau objek.

3.3 Instrumen Penelitian

Instrumen atau alat pengumpul data yang digunakan pada penelitian ini terdiri dari beberapa hal, di antaranya adalah komputasi yang digunakan, bahasa pemrograman yang digunakan, perangkat lunak dan *library* yang digunakan, serta metrik-metrik untuk mengevaluasi model pembelajaran mesin. Adapun untuk menetapkan fokus penelitian digunakan pendekatan *Goal Question Metric* (GQM) sebagai berikut:



Gambar 3.3 *Goal Question Metric* (GQM)

Pada penelitian ini, sebuah platform pembelajaran mesin dari Google Cloud Platform (GCP) yaitu Vertex AI digunakan sebagai komputasi untuk mengembangkan model pembelajaran mesin. Spesifikasi komputasi tersebut ditunjukkan oleh Tabel 3.4. Bahasa pemrograman yang digunakan untuk mengembangkan model pembelajaran mesin adalah bahasa pemrograman Python. Perangkat lunak dan beberapa *library* pendukung yang digunakan dalam penelitian ini ditunjukkan oleh Tabel 3.5. Kemudian untuk evaluasi model pembelajaran mesin digunakan beberapa metrik atau rumus evaluasi untuk permasalahan regresi yang ditunjukkan oleh Tabel 3.6.

Tabel 3.4

Spesifikasi Komputasi Vertex AI yang Digunakan

Tipe Mesin	n1-standard-4
vCPUs	4
Memory	15 GB
Storage	100 GB

Tabel 3.5

Perangkat Lunak dan *Library* yang Digunakan

No.	Nama	Deskripsi
1	Visual Studio Code (VS Code)	VS Code digunakan untuk editor kode program pembelajaran mesin yang dihubungkan dengan komputasi yang telah disebutkan sebelumnya menggunakan SSH.
2	Python 3.7.12	Bahasa pemrograman yang digunakan dalam pengembangan mesin pada penelitian adalah Python dengan versi 3.7.12.

3	Python Package Index (PIP) 22.3	PIP merupakan sistem pengelolaan paket (*package manager*) untuk bahasa pemrograman Python yang digunakan untuk menginstal beberapa *library* pendukung dalam penelitian ini. Versi yang digunakan adalah 22.3.
4	Jupyter Notebook	Jupyter Notebook yang digunakan adalah berasal dari ekstensi VS Code. Jupyter Notebook digunakan untuk menuliskan dan menjalankan kode pembelajaran mesin berbasis Python dengan ekstensi file ".ipynb".
5	Pandas	Pandas adalah *library* Python yang memiliki beragam fungsi dan struktur data yang berguna untuk memanipulasi dan mengolah data.
6	Numpy	Numpy adalah *library* Python yang digunakan untuk mengelola dan memanipulasi data dalam bentuk array dan matriks.
7	Matplotlib	Matplotlib adalah *library* Python yang digunakan untuk membuat plot grafik dan visualisasi data.
8	Seaborn	Seaborn adalah *library* Python yang digunakan untuk visualisasi data. Seaborn menyediakan fungsi yang lebih baik dari matplotlib.
9	Scikit-learn	Scikit-learn adalah *library* Python yang digunakan untuk membuat dan mengevaluasi model pembelajaran mesin. Beberapa algoritma dan fungsi pada scikit-learn digunakan dalam penelitian ini, seperti beberapa algoritma regresi, algoritma untuk seleksi fitur, fungsi untuk membagi dataset, validasi model, dan evaluasi model.
10	M5py	M5py adalah *library* Python yang digunakan untuk membuat model pembelajaran mesin dengan algoritma M5rules.
11	Tensorflow	Tensorflow adalah *library* Python yang digunakan untuk membuat model pembelajaran mesin. Tensorflow digunakan dalam penelitian ini untuk mengimplementasikan algoritma Artificial Neural Network (ANN).
12	Joblib	Joblib adalah *library* Python yang digunakan untuk menyimpan dan memuat objek Python. Pada penelitian ini joblib digunakan untuk menyimpan model pembelajaran mesin yang telah dilatih, sehingga dapat digunakan kembali di waktu yang akan datang tanpa perlu melatih ulah model tersebut.

Tabel 3.6

Metrik-Metrik Evaluasi Regresi

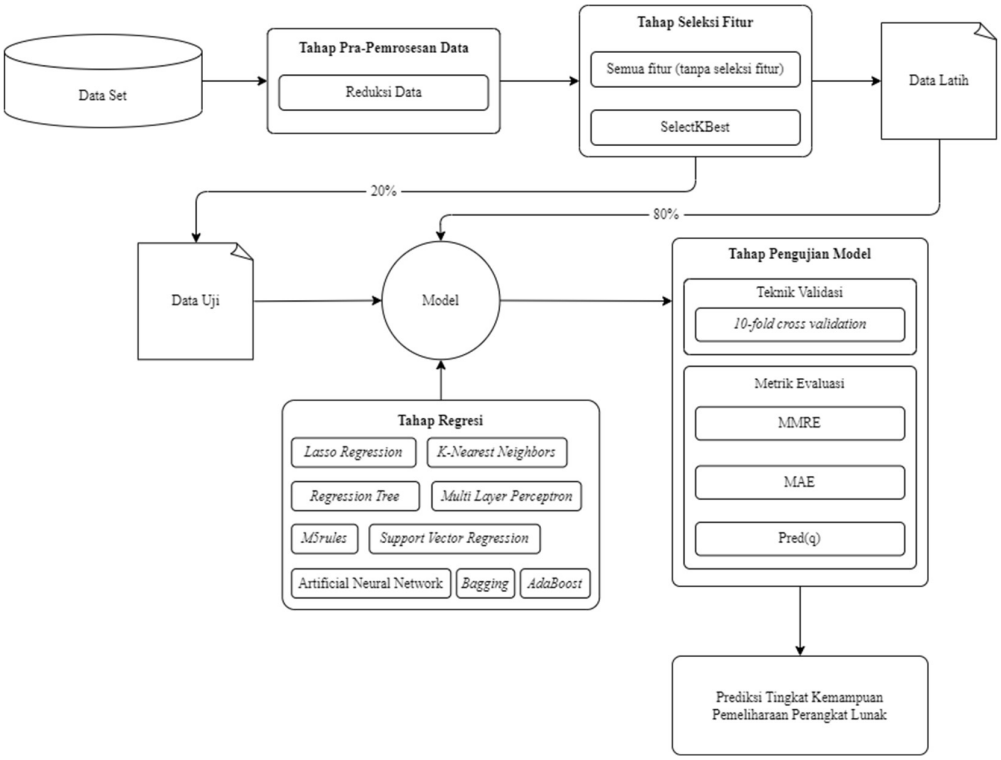
No.	Nama Metrik	Persamaan	Deskripsi
1	Magnitude of Relative Error (MRE)	$MRE_i = \frac{\lvert y_i - \hat{y}_i \rvert}{y_i}$	MRE menghitung selisih mutlak antara nilai aktual dan nilai prediksi dibagi dengan nilai aktual, di mana y_i adalah nilai aktual dan $\hat{y}_i$ adalah nilai prediksi.
2	Mean Magnitude of Relative Error (MMRE)	$MMRE = \frac{1}{n}\sum_{i=1}^{i=n} MRE_i$	Nilai rata-rata dari MRE
3	Mean Absolute Error (MAE)	$MAE = \frac{1}{n}\sum_{i=1}^{i=n} \lvert \hat{y}_i - y_i \rvert$	MAE menghitung nilai rata-rata dari selisih mutlak antara nilai prediksi dan nilai aktual, di mana y_i adalah nilai aktual dan $\hat{y}_i$ adalah nilai prediksi.
4	PRED (Pred(q))	$Pred(q) = \frac{k}{n}$	Pred menghitung proporsi dari *instance* dalam dataset di mana MRE kurang dari atau sama dengan nilai yang ditentukan (q). Variabel q didefinisikan sebagai nilai spesifik, k adalah jumlah di mana MRE kurang dari atau sama dengan q, dan n adalah seluruh jumlah dataset.

Penelitian ini akan mengevaluasi dan membandingkan model prediksi tingkat kemampuan pemeliharaan perangkat lunak pada sistem berorientasi objek secara kuantitatif, oleh karena itu pada Tabel 3.3 dapat diketahui bahwa penelitian ini menggunakan beberapa pengukuran akurasi untuk model prediksi seperti MMRE, MAE, dan Pred(q). Untuk pengukuran dengan MMRE dan MAE, semakin kecil nilai yang di dapat, maka semakin akurat model prediksi yang dihasilkan (Kumar dkk., 2019). Sedangkan untuk pengukuran Pred(q), semakin besar nilai yang di dapat, maka semakin akurat model prediksi yang dihasilkan (Van Koten & Gray, 2006). Pada penelitian ini akan menggunakan Pred(0.25) dan Pred(0.30) sebagai parameter, karena keduanya umum digunakan dalam penelitian terkait sebelumnya (Van Koten & Gray, 2006) (Zhou & Leung, 2007). Untuk mengetahui model

prediksi yang dibuat telah cukup akurat, terdapat standar minimal dari beberapa pengukuran, yaitu untuk pengukuran MMRE ≤ 0.25 dan atau Pred(0.25) ≥ 0.75 (Kitchenham dkk., 2001) atau Pred(0.30) ≥ 0.70 (MacDonell, 1997).

3.4 Prosedur Penelitian

Penelitian ini akan menggunakan kerangka kerja yang mengacu pada beberapa penelitian sebelumnya dan telah dimodifikasi untuk proses pembuatan model yang kemudian akan di evaluasi menggunakan beberapa rumus-rumus terkait untuk mengukur performa model. Kerangka kerja yang diusulkan dalam penelitian ini ditunjukkan oleh Gambar 3.4.



Gambar 3.4 Kerangka Kerja yang Diusulkan

Tahapan kerangka kerja yang diajukan ini secara umum terdiri dari tiga tahapan besar yaitu persiapan dataset, pembuatan model, dan pengujian model. Seluruh tahapan ini tentunya mengacu pada model DRM yang telah disampaikan pada bagian sebelumnya. Dalam tahapan persiapan dataset, dilakukan pra-pemrosesan data dan seleksi fitur yang dilakukan dengan 2 eksperimen. Eksperimen pertama menggunakan semua atribut yang ada atau tidak menggunakan seleksi fitur dan

eksperimen kedua dilakukan seleksi fitur menggunakan *SelectKBest*. Teknik seleksi fitur *SelectKBest* dipilih karena telah terbukti paling efektif dalam menghilangkan fitur yang tidak penting (Otchere dkk., 2022).

Tahapan kedua merupakan pembuatan model yang dimulai dengan membagi *dataset* menjadi data latih sebanyak 80% dan data uji sebanyak 20% secara acak. Kemudian digunakan beberapa algoritma untuk masalah regresi berdasarkan beberapa penelitian terdahulu (H. Alsolai & Roper, 2020) yang digunakan untuk pelatihan model dengan menggunakan data latih. Pada tahapan terakhir, pengujian dilakukan dengan teknik validasi berupa *10-fold cross-validation* pada saat pelatihan model untuk mengukur kemampuan generalisasi dari model yang dihasilkan. Kemudian, untuk mengevaluasi model yang dihasilkan, digunakan beberapa rumus untuk masalah regresi yang ditunjukkan pada Tabel 3.6. Pada tahap ini data uji digunakan untuk menguji performa model pembelajaran mesin.

3.5 Analisis Data

Seluruh hasil yang diperoleh dari setiap model akan dianalisis lebih lanjut dengan analisis komparatif untuk mendapatkan kesimpulan penelitian. Hasil performa model dari eksperimen pertama dibandingkan dengan penelitian terdahulu. Selanjutnya hasil eksperimen kedua yang menggunakan seleksi fitur akan dibandingkan dengan hasil eksperimen pertama. Adapun perangkat lunak pendukung yang digunakan untuk analisis data adalah Microsoft Excel dan bahasa pemrograman Python dengan *library matplotlib* dan *seaborn*. *Library* matplotlib dan *seaborn* digunakan untuk membuat plot grafik dan visualisasi data.