**Pendahuluan**

Perkiraan *effort* dalam pengembangan perangkat lunak adalah aspek yang sangat penting dalam manajemen proyek perangkat lunak. Estimasi ini membantu para manajer proyek dalam merencanakan sumber daya, biaya, dan jadwal pengembangan. Penelitian ini bertujuan untuk memperkirakan effort dengan menggunakan model machine learning, yang memungkinkan prediksi yang lebih tepat dan efisien dibandingkan dengan metode konvensional. Dataset yang digunakan dalam eksperimen ini adalah NASA93, yang berisi data dari 93 proyek perangkat lunak yang dikembangkan di NASA. Dataset ini terdiri dari 93 baris dan 50 kolom, dengan fitur-fitur yang relevan seperti ukuran tim, kompleksitas proyek, reliabilitas perangkat lunak, dan waktu yang dibutuhkan untuk proyek. Data ini digunakan untuk melatih model machine learning guna memprediksi *effort* dalam pengembangan perangkat lunak. Penelitian ini untuk membandingkan performa beberapa algoritma machine learning dalam memperkirakan *effort* pengembangan perangkat lunak. Model-model yang diuji termasuk K Neighbors Regressor, Random Forest Regressor, Linear Regression, Support Vector Regression, dan Decision Tree Regresso. Selain itu, penelitian ini untuk mengeksplorasi pengaruh variabel independen terhadap *effort* yang dibutuhkan, serta menilai akurasi dan kemampuan model dalam menjelaskan variasi data. Dalam eksperimen ini, data dibagi menjadi set pelatihan (70%) dan set pengujian (30%), dengan pembagian ini memastikan evaluasi yang lebih objektif tentang performa model. Metrik pengukuran kesalahan seperti MAE, MSE, RMSE, R2, RMSLE dan MAPE digunakan untuk mengevaluasi kinerja model yang diuji. Hasil dari eksperimen ini memberikan wawasan tentang bagaimana machine learning dapat digunakan untuk meningkatkan akurasi *effort estimation* dalam pengembangan perangkat lunak, yang dapat membantu manajer proyek untuk membuat keputusan yang lebih baik terkait perencanaan dan pengelolaan sumber daya.

**Metode Estimasi Effort**

* Penilaian Ahli ***(Expert Judgment)*** *:* Bergantung pada pengetahuan individu yang pernah terlibat dalam proyek yang sejenis. Teknik ini menghasilkan estimasi awal tetapi dapat terpengaruh oleh bias dan dipengaruhi oleh pengalaman pribadi.
* Estimasi Analog ***(Analogous Estimation)*** *:* Memanfaatkan informasi dari proyek-proyek yang telah selesai untuk menilai *effort* dari proyek yang sedang dijalankan. Teknik ini relatif cepat dan sederhana, tetapi bisa kurang tepat karena proyek yang dijadikan acuan mungkin memiliki perbedaan.
* Estimasi Tiga Titik ***(Three-Point Estimation)*** *:* Dengan menerapkan perkiraan optimis, pesimis, dan yang paling mungkin untuk mendapatkan kisaran nilai *effort*. Pendekatan ini menawarkan pandangan yang lebih luas meskipun lebih rumit.
* Estimasi Parametrik ***(Parametric Estimation)*** *:* Memanfaatkan model matematika yang didasarkan pada skala, tingkat kesulitan, serta variabel lainnya dari proyek. Penilaian ini menawarkan hasil yang lebih tepat meskipun membutuhkan data yang lebih lengkap dan pemahaman yang lebih baik mengenai proyek tersebut.

Estimasi *effort* dalam pengembangan software merupakan elemen krusial dalam pengelolaan proyek perangkat lunak, yang membantu dalam merencanakan anggaran, sumber daya, dan *timeline* yang dibutuhkan. Kesalahan dalam estimasi yang terjadi di fase awal proyek sering kali menjadi faktor utama penyebab kegagalan proyek perangkat lunak. Penelitian ini bertujuan untuk mengevaluasi apakah machine learning dapat berfungsi sebagai metode yang lebih efektif dibandingkan dengan pendekatan tradisional dalam memperkirakan *effort* pengembangan perangkat lunak. Dalam penelitian ini, berbagai algoritma machine learning, termasuk K Neighbors Regressor, Random Forest Regressor, Linear Regression, Support Vector Regression, dan Decision Tree Regressor, algoritma regresi dan regresi berbasis pohon keputusan diterapkan, dengan dukungan dari bahasa pemrograman Python.

**Without Machine Learning**

COCOMO adalah sebuah model yang mengandalkan rumus dan parameter berdasarkan pengalaman dari proyek-proyek sebelumnya untuk memperkirakan biaya pengembangan perangkat lunak. Model ini banyak digunakan karena kesederhanaannya, meskipun ada pengembangan lebih lanjut untuk meningkatkan akurasi perhitungan. Salah satu pengembangan fuzzy-COCOMO ditambahkan untuk meningkatkan ketepatan estimasi dengan mengintegrasikan Gaussian Membership Function, yang menjadikannya lebih berguna dalam memperkirakan biaya. Selain itu, metode baru juga diterapkan dengan menggunakan algoritma BATGSA untuk mengoptimalkan COCOMO, memanfaatkan data dari berbagai sumber, seperti database NASA, yang berhasil menurunkan kesalahan ter-normalisasi secara signifikan, serta meningkatkan ketepatan estimasi *effort* dengan menggabungkan teknik seperti *fuzzy* clustering, ABE, dan ANN untuk perhitungan yang lebih akurat.

**Machine Learning**

Machine Learning digunakan untuk memprediksi *effort* dalam pengembangan perangkat lunak, memanfaatkan dataset NASA93 yang meliputi 93 proyek perangkat lunak dan 50 kolom fitur penting seperti ukuran tim, kerumitan proyek, serta keandalan, di mana data dipecah menjadi 70% untuk pelatihan dan 30% untuk pengujian. Evaluasi dilakukan dengan metrik MAE, MSE, RMSE, R², RMSLE, dan MAPE. Hasil menunjukkan bahwa Regresi Linier (LR) memberikan prediksi paling tepat dengan MAE dan RMSE terendah, sementara Regressor Random Forest memiliki nilai R² terbaik, menunjukkan kemampuannya yang superior dalam menangkap variasi data. Oleh karena itu, penelitian ini menyimpulkan bahwa pembelajaran mesin mampu meningkatkan estimasi *effort* dalam pengembangan perangkat lunak dan memberikan pencerahan bagi manajer proyek untuk perencanaan sumber daya dan waktu yang lebih efisien.

**K Neighbors Regressor**

K Neighbors Regressor merupakan metode non-parametrik yang melakukan prediksi dengan mengandalkan nilai dari titik data tetangga. Algoritma ini melibatkan dua jenis variabel, yaitu variabel dependen dan variabel independen. Algoritma memprediksi variabel dependen dengan menggunakan variabel independen sebagai acuan untuk melakukan prediksi. KNN beroperasi dengan menemukan K titik data terdekat dari data yang diberikan dan menghasilkan prediksi berdasarkan rata-rata (untuk regresi) atau mayoritas (untuk klasifikasi) dari tetangga terdekat. Berbeda dari regresi linier, KNN tidak menetapkan asumsi mengenai bentuk distribusi data tertentu. Kompleksitas model bergantung pada jumlah tetangga yang dianggap dan metrik jarak yang dipilih. KNN sangat efektif untuk dataset di mana hubungan antar variabel bersifat *non-linear* dan mampu menghadapi struktur data yang kompleks. Metode ini mudah untuk diterapkan dan dimengerti, menjadikannya alat yang relatif ringan dan fleksibel untuk tugas klasifikasi dan regresi.

**Random Forest Regressor**

Random Forest Regressor digunakan untuk menciptakan decision tree, yang dapat diterapkan pada isu klasifikasi maupun regresi. Selanjutnya untuk memperluas metode ini dengan menyatukan *bagging* dan pemilihan fitur secara acak, sehingga memungkinkan pembuatan beberapa decision trees secara terstruktur sambil menggunakan variasi. Jika dibandingkan dengan satu decision tree, algoritma Random Forest menawarkan perkiraan tingkat kesalahan yang lebih tepat. Terutama, bukti matematis menunjukkan bahwa tingkat kesalahan cenderung berkurang seiring dengan bertambahnya jumlah pohon dalam Random Forest. Secara umum, karena kemampuan Random Forest untuk beradaptasi dengan ketidaklinieran yang terdeteksi dalam data, model ini seringkali memberikan performa yang lebih baik dibandingkan regresi linier.

**Linear Regression**

Linear Regression merupakan metode yang diaplikasikan untuk memprediksi nilai dari variabel yang bergantung berdasarkan nilai dari variabel lainnya yang tidak bergantung. Dalam pendekatan ini, terdapat dua tipe variable satu variabel yang bergantung yang diprediksi oleh metode ini dan satu variabel yang tidak bergantung sebagai dasar prediksi. Analisis ini menemukan nilai-nilai koefisien dalam persamaan matematis linier dengan memanfaatkan variabel tidak bergantung yang paling efisien untuk memprediksi nilai dari variabel yang bergantung. Metode ini menyesuaikan hasil dalam bentuk garis lurus untuk meminimalkan selisih antara nilai sebenarnya dan nilai yang diprediksi. Nilai A (variabel yang bergantung) kemudian diprediksi dari B (variabel yang tidak bergantung). Metode ini cukup sederhana dan dalam perbandingan, dianggap lebih mudah dibandingkan teknik prediksi yang lain. Persamaan untuk Linear Regression dapat dinyatakan sebagai berikut:

*Y = β0 + β1* *X1 + β2* *X1 +*… + *βn* *Xn* + e

Dimana *Y* merupakan variabel hasil, 𝛽*1*, 𝛽2, …, 𝛽*𝑛* adalah nilai koefisien yang diperoleh melalui perhitungan, dan X\_1, X\_2, ..., X\_n adalah variabel yang tidak tergantung. Pada bagian ini, 𝛽0 adalah titik potong dari garis yang ditentukan oleh sumbu *Y,* dan *e* merepresentasikan salah satu jenis kesalahan acak yang bertujuan menunjukkan dampak faktor-faktor acak terhadap *Y* (variabel hasil).

**Support Vector Regression**

Support Vector Regression merupakan teknik regresi yang berasal dari Support Vector Machine. Teknik ini memanfaatkan prinsip Minimasi Risiko Struktural untuk mengurangi penyimpangan dari nilai asli serta mempertahankan kemampuan generalisasi. Ada dua *hyperparameter* utama dalam SVR, yaitu fungsi kernel yang berfungsi untuk memproyeksikan data ke dalam ruang berdimensi tinggi dan parameter C yang menentukan toleransi terhadap kesalahan. Pemilihan kernel yang tepat, seperti *linear, polinomial,* atau RBF, memiliki dampak besar pada kinerja model. Hal ini menjadikan SVR sangat berguna dalam menangani hubungan *non-linear* dan sering digunakan untuk memperkirakan upaya dalam proyek perangkat lunak.

xi w + b = 0

x*i* w + *b* ≥ + 1, y*i = +* 1

x*i* w + *b* ≤ – 1, y*i =* –1

y*i* (x*i* w + *b*) – 1 ≥ 0, *i*

minimize

Maximize

**Decision Tree Regressor**

Decision Tree Regression merupakan teknik yang berlandaskan pada struktur pohon untuk memprediksi nilai numerik dari variabel yang bergantung, menggunakan algoritma M5P yang menyerupai struktur pohon CART *(Classification And Regression Trees).* Di dalam struktur yang dihasilkan oleh M5P, terdapat model linier multivariat, sementara pohon regresi menampilkan nilai pada bagian daun. Proses pembentukan pohon ini dilakukan dengan menerapkan kriteria pemisahan yang bertujuan untuk mengurangi variasi nilai kelas di masing-masing cabang, menggunakan rumus penurunan standar deviasi untuk menghitung hasil pemisahan antara *node-node.*

Setelah algoritma machine learning diterapkan pada dataset, lima indikator statistik utama, yaitu *Mean Absolute Error, Mean Squared Error, Root Mean Squared Error, Correlation Coefficient, Root Mean Squared Logarithmic Error, dan Mean Absolute Percentage Error,* digunakan untuk mengukur kesalahan antara prediksi dan nilai aktual dari effort proyek, dengan menghitung rata-rata kesalahan absolut, kuadrat kesalahan, akar kuadrat kesalahan, persentase kesalahan, serta seberapa baik model menjelaskan variabilitas data, yang memungkinkan evaluasi kinerja dan akurasi model.

**Metode**

**Data Pre-Processing**

Data yang digunakan berasal dari dataset NASA93 yang berisi 93 proyek pengembangan perangkat lunak di NASA. Dataset ini terdiri dari 93 baris dan 50 kolom, dengan fitur seperti ukuran tim, kompleksitas, reliabilitas, dan *effort* yang dibutuhkan. Nilai nominal diubah menjadi nilai numerik yang dapat dibandingkan, dan kolom deskriptif yang tidak relevan, seperti nama proyek dan kategori proyek, dihapus. Nilai yang hilang diimputasi menggunakan rata-rata kolom yang bersangkutan. Dataset kemudian diproses ke dalam format CSV untuk mempermudah pelatihan model. Tabel 1 memberikan deskripsi lengkap dari atribut yang digunakan dalam eksperimen.

Tabel 1 Deskripsi Dataset

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Symbol** | **Description** | **Datatype** |
| Project Effort | Project Name | Name | String |
| Application Experience | apex | Experience in the specific application domain | String |
| Communication Infrastructure | Communication Infrastructure | Communication systems used by the project team | Character [f , g] |
| Team Size | team | Increase these to decrease effort | Positive integer |
| Required Software Reliability | rely |
| Database Size | data |
| Complexity | cplx |
| Reuse | ruse |
| Documentation | docu |
| Time Restriction | time |
| Storage Restriction | stor |
| Programming Volume | pvol |
| Analyst Capability | acap | Increase these to decrease effort | Positive integer |
| Programmer Capability | pcap |
| Project Constraints | pcon |
| Process Experience | plex |
| Language Experience | ltex |
| Tool Usage | tool |
| Schedule Constraints | sced |
| Lines of Code | kloc |
| Development Effort | effort | Total effort required in terms of person-months | Positive integer |

Tabel 2 Parameter GridSearchCV untuk SVR

|  |  |
| --- | --- |
| Variable | Value |
| C | 10 |
| Gamma | Auto |
| Karnel | Linear |

**Model Training**

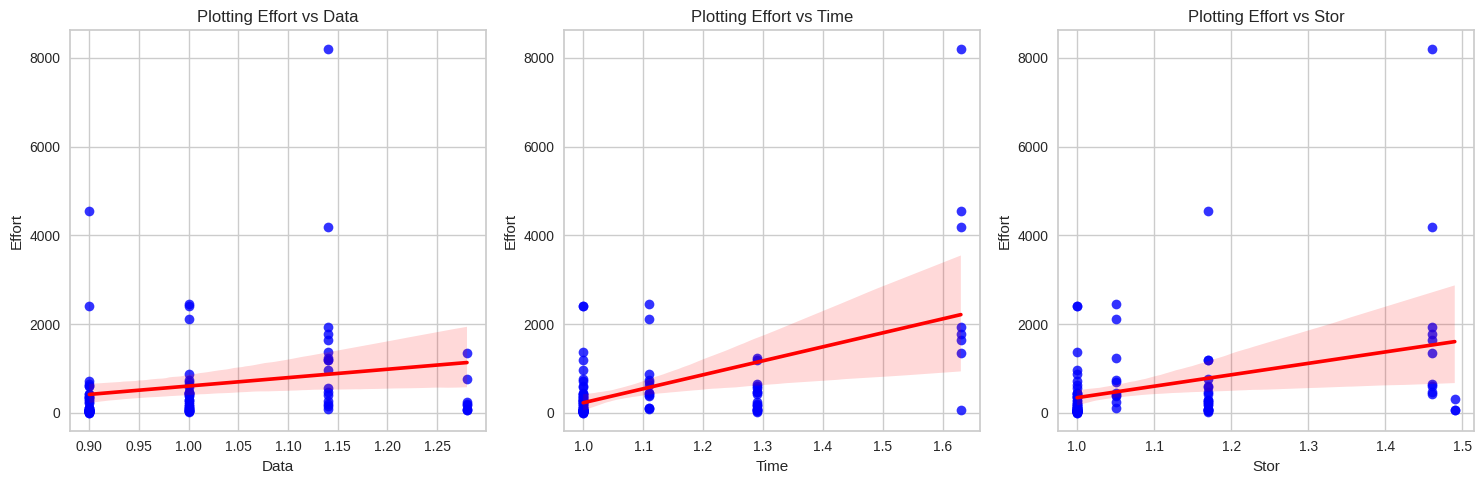
“Effort” adalah variabel dependen utama dalam eksperimen ini. Nilai aktual dari dataset akan dibandingkan dengan nilai yang diprediksi sebagai bagian dari proses prediksi yang dilakukan oleh algoritma Machine Learning. Dataset yang digunakan sebagian besar terdiri dari data numerik, dengan lebih dari 70% data berbentuk numerik. Selanjutnya, dataset dibagi secara acak menjadi set pelatihan dan set pengujian dengan rasio 70:30, yang berarti 70% dari data digunakan untuk melatih model, dan 30% sisanya digunakan untuk menguji keakuratan model. Pembagian ini memungkinkan model untuk dilatih pada sebagian besar data dan diuji pada data yang belum pernah dilihat sebelumnya, memastikan evaluasi yang lebih objektif mengenai performa model.

**Hasil**

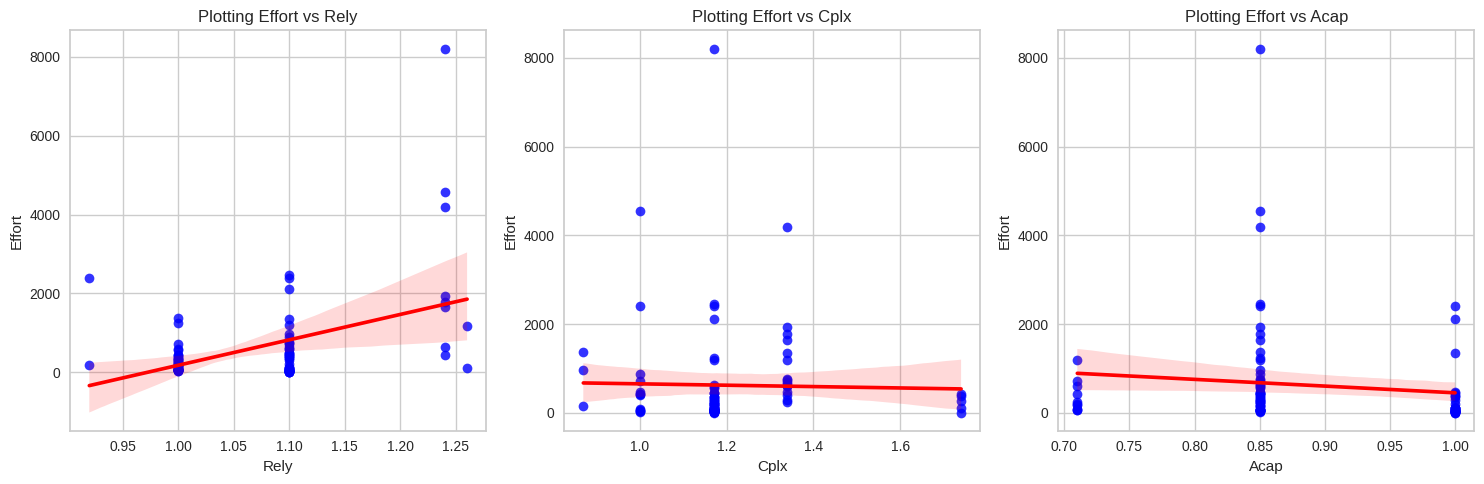
Hasil eksperimen yang dilakukan untuk memperkirakan software effort dengan menggunakan lima model machine learning. Model-model yang diuji meliputi K Neighbors Regressor, Random Forest Regressor, Linear Regression, Support Vector Regression, dan Decision Tree Regressor. Hasil eksperimen ini menunjukkan perbedaan kinerja model dalam memprediksi *effort* proyek perangkat lunak. Gambar 1 hingga Gambar 3 menunjukkan hubungan antara variabel dependen *(effort)* dan variabel independen yang relevan dalam dataset NASA93. Gambar 1 menunjukkan hubungan antara *effort* dan data, di mana terdapat kecenderungan hubungan positif meskipun tidak sangat signifikan. Gambar 2 menggambarkan hubungan antara *effort* dan *time*, dengan hubungan yang lebih jelas tetapi tetap lemah. Gambar 3 memperlihatkan hubungan antara *effort* dan *stor,* di mana hubungan positif terlihat dengan intensitas yang lebih rendah.

Tabel 3 hasil evaluasi dari lima model yang diuji menggunakan metrik MAE, MSE, RMSE, R², RMSLE, dan MAPE. Hasil menunjukkan bahwa Linear Regression memiliki performa terbaik dalam hal MAE (253.48) dan RMSE (356.86), yang menunjukkan bahwa model ini lebih akurat dalam memprediksi *effort* dibandingkan dengan model lainnya. Meskipun LR menunjukkan hasil terbaik dalam hal MAE dan RMSE, Random Forest Regressor memberikan hasil terbaik dalam hal R², dengan nilai 0.1146. Ini menunjukkan bahwa RF lebih efektif dalam menjelaskan variabilitas data meskipun memiliki MAE dan RMSE yang sedikit lebih tinggi. Sebaliknya, Support Vector Regression dan Decision Tree Regressor memberikan hasil yang kurang baik, dengan MAE dan RMSE yang lebih tinggi serta R² yang sangat negatif untuk DT (-3.8918), yang menunjukkan bahwa model ini gagal untuk menjelaskan pola dalam data.

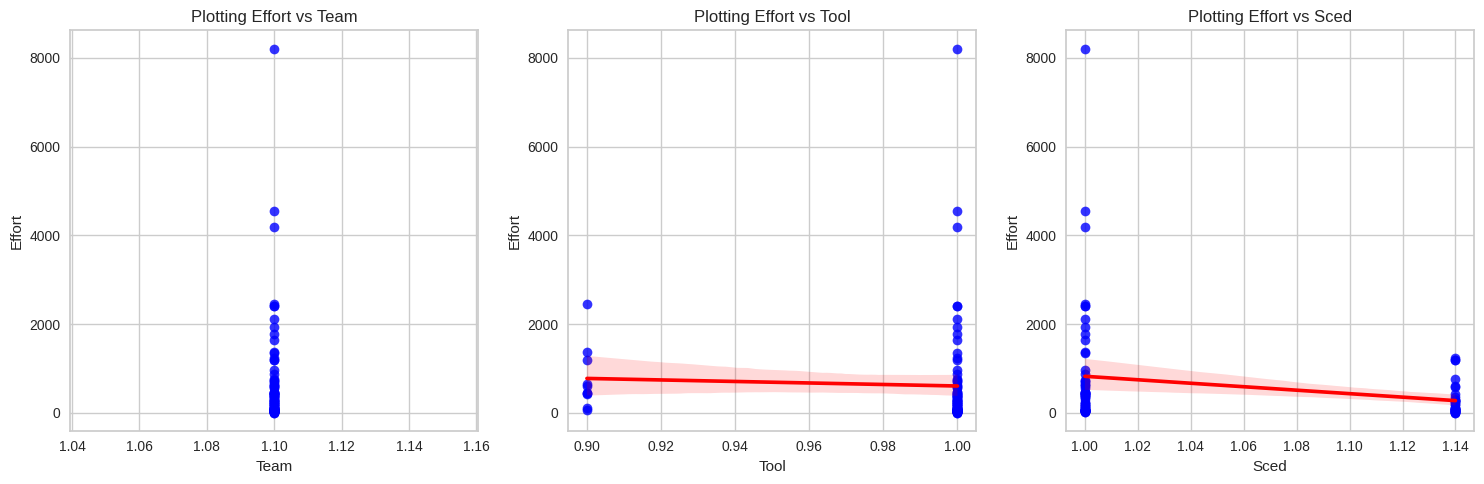
Berdasarkan hasil eksperimen, Linear Regression lebih unggul dalam hal akurat prediksi *effort* dengan MAE dan RMSE terendah. Namun, Random Forest Regressor memiliki R² terbaik, yang menunjukkan bahwa model ini lebih baik dalam menangkap variasi dalam data meskipun tidak sebaik LR dalam hal prediksi yang akurat. Model-model seperti SVM dan DT menunjukkan kinerja yang buruk dalam memprediksi *effort,* terutama karena MAE dan RMSE lebih tinggi serta R² yang sangat negatif untuk DT. KNN menunjukkan MAPE yang lebih tinggi, meskipun hasilnya tidak buruk dalam beberapa metrik.



Gambar 1 Plotting Variabel Dependen (Effort) vs. Variabel Independen



Gambar 2 Plotting Variabel Dependen (Effort) vs. Variabel Independen



Gambar 3 Plotting Variabel Dependen (Effort) vs. Variabel Independen

Tabel 3 Hasil model ML Menggunakan Indeks Pengukuran Kesalahan

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model | MAE | MSE | RMSE | R2 | RMSLE | MAPE |
| K Neighbors Regressor | 326.7018 | 390045.3765 | 509.6352 | 0.1360 | 0.9885 | 1.1252 |
| Random Forest Regressor | 265.3411 | 313067.5210 | 472.3661 | 0.1146 | 0.7038 | 0.9022 |
| Linear Regression | 253.4827 | 153644.4004 | 356.8596 | -0.0789 | 0.9259 | 1.8475 |
| Support Vector Regression | 430.1797 | 592965.2850 | 645.6916 | -0.2733 | 1.4939 | 4.0474 |
| Decision Tree Regressor | 359.7562 | 584455.7254 | 643.6188 | -3.8918 | 0.8476 | 0.7554 |

**Kesimpulan**

Dalam penelitian ini lima model machine learning, yaitu K Neighbors Regressor, Random Forest Regressor, Linear Regression, Support Vector Regression, dan Decision Tree Regressor, diuji untuk memperkirakan *software effort* menggunakan dataset NASA93, dengan hasil yang menunjukkan bahwa Linear Regression memberikan hasil terbaik dalam hal akurasi prediksi dengan MAE (253.48) dan RMSE (356.86) terendah, meskipun Random Forest Regressor memiliki R² terbaik (0.1146) yang menunjukkan kemampuan model ini dalam menjelaskan variabilitas data meskipun memiliki MAE dan RMSE yang sedikit lebih tinggi, sementara SVM dan DT menunjukkan kinerja yang buruk dengan MAE dan RMSE yang lebih tinggi serta R² yang sangat negatif untuk DT (-3.8918), dan KNN juga memberikan hasil yang kurang optimal dengan MAPE yang lebih tinggi, sehingga secara keseluruhan Linear Regression lebih unggul dalam prediksi akurat dan Random Forest Regressor lebih baik dalam menjelaskan variasi data, dengan SVM dan DT tidak disarankan, serta penelitian selanjutnya dapat mengeksplorasi XGBoost atau Neural Networks untuk meningkatkan akurasi dan pemahaman terhadap variasi data dalam *effort* estimation.

**Referensi**