SKRIPSI

IMPLEMENTASI LSSVM DALAM KLASIFIKASI DATA PREDIKSI CACAT PERANGKAT LUNAK DENGAN *FEATURE SELECTION*

Oleh:

Thingkilia Finnatia Husin **1822250026**

PROGRAM STUDI INFORMATIKA
FAKULTAS ILMU KOMPUTER DAN REKAYASA
UNIVERSITAS MULTI DATA PALEMBANG
PALEMBANG

2022

Fakultas Ilmu Komputer dan Rekayasa
Universitas Multi Data Palembang

Program Studi Informatika
Skripsi Sarjana Komputer
Semester Genap Tahun 2021/2022

IMPLEMENTASI LSSVM DALAM KLASIFIKASI DATA PREDIKSI CACAT PERANGKAT LUNAK DENGAN *FEATURE SELECTION*

Thingkilia Finnatia Husin 1822250026

Abstrak

Biaya pengembangan industri TI secara global terus meningkat setiap tahun yang sekitar 35% pengeluaran digunakan untuk biaya *testing* dalam jaminan kualitas dan pengujian. Prediksi cacat perangkat lunak merupakan upaya pendekatan yang dilakukan untuk meningkatkan kualitas, efisiensi, dan efektivitas waktu serta biaya dalam pengujian perangkat lunak dengan berfokus pada modul cacat. Dengan teknologi prediksi cacat perangkat lunak dapat memprediksi modul cacat secara cepat dengan metode *machine learning* sehingga mempermudah dalam alokasi sumber daya yang terbatas. *Dataset* prediksi cacat perangkat lunak secara alami memiliki masalah ketidakseimbangan kelas dengan modul yang cacat / *defect* sangat sedikit dibandingkan dengan modul yang tidak cacat. Pada penelitian ini dilakukan klasifikasi data prediksi cacat perangkat lunak / *software defect prediction* dengan implementasi algoritma LSSVM disertai ReliefF (K=10) *feature selection* dan menerapkan metode SMOTE untuk mengatasi masalah *imbalance class* pada dataset. *Dataset* yang digunakan berupa *dataset* prediksi cacat perangkat lunak proyek NASA MDP Promise yang bersifat *public*, yaitu CM1, KC1, KC2, dan PC1. Pembagian *dataset* menjadi data *training* dan *testing* menggunakan *Fold Cross Validation* dengan 10 *fold*. Hasil akurasi rata – rata tertinggi dicapai *classifier* pada dataset PC1, yaitu sebesar 93,87%, sedangkan nilai *Area Under the ROC Curve (AUC)* tertinggi dicapai oleh *classifier* untuk dataset KC2, yaitu sebesar 78,35%.

Kata kunci: Klasifikasi, LSSVM, ReliefF, SMOTE, *Software Defect Prediction*.


UNIVERSITAS
MDP

BAB 1

PENDAHULUAN

Bab ini menjelaskan tentang latar belakang mengapa penelitian ini dilakukan, perumusan masalah yang akan diselesaikan, ruang lingkup pengerjaan, tujuan dan manfaat yang akan diperoleh, metodologi penelitian, serta sistematika penulisan laporan penelitian.

1.1 Latar Belakang

Teknologi Informasi berkembang sangat pesat dan kebutuhannya semakin meningkat seiring berjalannya waktu. Sistem perangkat lunak yang dibangun menjadi semakin lebih kompleks. Akibatnya, hal ini meningkatkan biaya dalam proses pengujian perangkat lunak secara menyeluruh. Berdasarkan data *research* yang dilakukan oleh Justina Alexandra Sava selaku *Research Expert Covering The IT Security & Services Market*, pengeluaran teknologi informasi (TI) global mencapai $3,8 triliun pada tahun 2020 dan diperkirakan akan meningkat sekitar 5,1% menjadi sekitar $4,45 triliun pada 2022 (Sava, 2022). Sekitar 35% dari total biaya pengembangan di industri TI, biaya *testing* dihabiskan dalam jaminan kualitas dan pengujian. (Kulamala et al., 2021)

Upaya untuk mengurangi biaya pengujian perangkat lunak tersebut dapat dengan melakukan prediksi cacat perangkat lunak. Prediksi cacat perangkat lunak

adalah pendekatan praktis untuk meningkatkan kualitas, efisiensi dan efektivitas waktu dan biaya untuk pengujian perangkat lunak dengan berfokus pada modul cacat (Bahaweres et al., 2020). Dengan teknologi prediksi cacat perangkat lunak dapat memprediksi modul cacat secara cepat dengan metode *machine learning* (Bahaweres et al., 2020). Modul cacat yang berhasil diprediksi akan mempermudah dalam alokasi sumber daya yang terbatas dengan memprioritaskan modul-modul yang diperkirakan mengandung cacat (Qiao et al., 2020).

Penelitian sebelumnya tentang analisis performa berbagai teknik klasifikasi *machine learning* dalam mengklasifikasi data prediksi cacat perangkat lunak dari NASA *Datasets* mengalami kendala berupa ketidakseimbangan kelas / *imbalance class* (Iqbal et al., 2019). Teknik klasifikasi yang digunakan pada penelitian tersebut, meliputi *Naïve Bayes* (NB), *Multi-Layer Perceptron* (MLP), *Radial Basis Function* (RBF), *Support Vector Machine* (SVM), *K-Nearest Neighbor* (KNN), *kStar* (K*), *One Rule* (OneR), *PART*, *Decision Tree* (DT), dan *Random Forest* (RF). Performa klasifikasi dievaluasi menggunakan *confusion matrix*. Namun, terjadi kendala ketidakseimbangan data yang menyebabkan terjadinya bias saat mengukur dan mengevaluasi hasil penelitian.

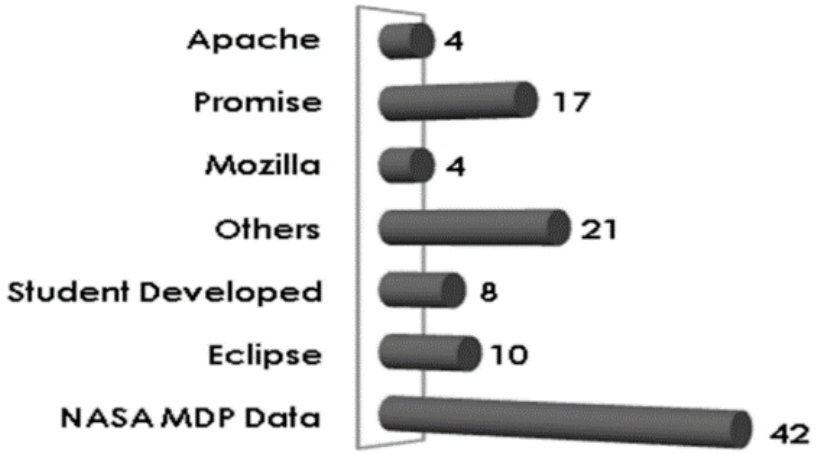
Dataset prediksi cacat perangkat lunak secara alami memiliki masalah ketidakseimbangan kelas dengan modul yang cacat sangat sedikit dibandingkan dengan modul yang tidak cacat (Bahaweres et al., 2020). Hal ini menyebabkan algoritma klasifikasi akan cenderung bias terhadap modul yang tidak cacat (kelas mayoritas), dan dalam kasus ekstrim dapat mengabaikan modul yang cacat (kelas

minoritas) sehingga menyebabkan *overfitting* dan performa yang buruk (Bahaweres et al., 2020). Untuk menangani masalah *imbalance class* pada *dataset* prediksi cacat perangkat lunak projek NASA, dapat dengan menerapkan *data-level method* dalam memanipulasi data training yang bertujuan mendistribusikan dataset menjadi lebih seimbang (Douzas et al., 2018). Metode yang dapat diterapkan dapat berupa *undersampling* dan *oversampling.* Metode *undersampling* melakukan manipulasi dataset dengan menghapuskan beberapa data pada kelas mayoritas, sedangkan metode *oversampling* melakukan penambahan *instance* baru pada kelas minoritas dengan cara menduplikasi atau pembuatan sampel baru (Douzas et al., 2018). Karena *undersampling* melakukan penghapusan beberapa data, metode tersebut berisiko kehilangan konsep / data penting. Selain itu, ketika jumlah kelas yang minoritas sangat sedikit, teknik *undersampling* dalam menyeimbangkan dataset dapat menyebabkan ukuran dataset yang semulanya besar menjadi sangat kecil sehingga dapat membatasi ataupun menurunkan kinerja *classifier* (Douzas et al., 2018). Salah satu solusi yang dapat menangani kasus *imbalanced* dengan metode *oversampling* adalah *Synthetic Minority Over-sampling Technique* (SMOTE).

Pada penelitian (Bahaweres et al., 2020) menggunakan metode SMOTE untuk mengatasi masalah *imbalance class* dan menggunakan algoritma *Neural Network*. Hasil penelitian (Bahaweres et al., 2020) menunjukkan bahwa terjadi peningkatan sebesar 25,48% pada nilai *balance (Bal)* dan 45,99% pada nilai *Recall* dibandingkan performa saat hanya menggunakan *Neural Network* tanpa SMOTE. Hal ini menunjukkan SMOTE berhasil mengatasi masalah *imbalance class* sehingga

meningkatkan performa. Penelitian (Bahaweres et al., 2020) juga membandingkan performa algoritma *Neural Network* + SMOTE dengan 5 algoritma *machine learning* lainnya, yaitu algoritma Random Forest, KNN, *Logistic Regression*, *Decision Tree*, dan *Naive Bayes* yang juga diaplikasikan dengan SMOTE. Berdasarkan *Average Rank Bal*, algoritma *Neural Network* + SMOTE mengungguli performa dari algoritma lainnya.

Untuk membangun model prediksi yang akurat, subset fitur yang relevan harus dipertimbangkan dengan cermat sebagai masukan untuk algoritma klasifikasi. Dalam hal prediksi cacat perangkat lunak, dataset NASA MDP (*Metrics Data Program*) adalah dataset populer yang telah digunakan dalam banyak penelitian karena berdasarkan *research* dan studi literatur yang pernah dilakukan Romi Satria Wahono, dataset ini paling bagus dan siap untuk digunakan oleh metode - metode klasifikasi (Khadijah et al., 2020). Dari 98 studi tentang prediksi cacat perangkat lunak (DeP / *Defect Prediction*) pada tahun 1995 hingga 2018, terdapat 42 penelitian yang menggunakan NASA Dataset sebagai dataset yang digunakan dalam penelitian yang ditunjukkan pada Gambar 1.1 (Son et al., 2019).



Gambar 1.1 Penggunaan *Dataset* pada penelitian prediksi cacat perangkat lunak (Son et al., 2019)

Dataset berisi sejumlah *static software metrics* sebagai atribut untuk menentukan adanya cacat dalam modul. Namun, belum diketahui apakah semua atribut relevan untuk memprediksi adanya cacat pada modul perangkat lunak (Wei et al., 2019). Menyertakan fitur yang tidak relevan dan berlebihan untuk membangun model pengklasifikasi dapat menurunkan performa model yang dihasilkan (Ghotra et al., 2017; Khadijah et al., 2020). Oleh karena itu fitur yang tidak relevan atau berlebihan tersebut harus dieliminasi dengan menggunakan algoritma *feature selection* (Khadijah et al., 2020). Seleksi fitur juga dapat meningkatkan kemampuan generalisasi *classifier* dan memberikan intuisi tentang fitur yang paling informatif. Para peneliti setuju bahwa pendekatan *feature selection* efektif dalam *high-dimensionality problems* (Balogun et al., 2021).

Penelitian sebelumnya (Khadijah et al., 2020) telah membandingkan beberapa metode *feature selection* pada klasifikasi dataset prediksi cacat perangkat lunak NASA MDP menggunakan metode SVM (*Support Vector Machine*). Metode *feature selection* yang dibandingkan adalah *Recursive Feature Elimination* (RFE) , ReliefF (K=5), dan ReliefF (K=10). *Dataset* yang digunakan yaitu PC1, PC2, PC3, dab PC4 *dataset*. Hasil penelitian ini mengatakan bahwa penggunaan SVM dengan ReliefF (K=10) memiliki nilai akurasi tertinggi. Pada penelitian (Khadijah et al., 2020) mencobakan beberapa jumlah fitur yang dipilih, mulai dari 10 hingga jumlah fitur di setiap set data untuk mendapatkan model terbaik, dimana hasil dengan akurasi terbaik pada metode ReliefF(K=10) didapatkan pada dataset PC2 dengan K=10 dan banyak fitur yang dipillih adalah 27.

SVM merupakan algoritma yang sering digunakan dalam *machine learning* dan implementasi pengenalan pola (Kulamala et al., 2021). SVM mampu memisahkan data *training* ke dalam kelas dengan membangun *decision surface* yang dibatasi oleh *support vectors*. SVM lebih disukai daripada teknik pembelajaran mesin lainnya dikarenakan memiliki tingkat akurasi prediksi yang lebih tinggi. Namun, penggunaan SVM sering menyebabkan biaya komputasi yang tinggi (Kulamala et al., 2021). Algoritma LSSVM (*Least Square Support Vector Machine*) mampu menurunkan biaya komputasi tersebut dengan mengurangi permasalahan yang menjadikannya sebagai sistem linear (Kulamala et al., 2021; Kumar et al., 2018). Pada dasarnya, metode LSSVM dapat memecahkan masalah yang dapat dipisahkan oleh *optimal hyperplane* yang dibedakan dengan sejumlah *support vector*. LSSVM mereduksi

SVM menjadi sistem linier . Oleh karena itu, LSSVM dapat sangat berguna dalam masalah optimasi, dimana hal itu sangat penting dalam melakukan klasifikasi (Kulamala et al., 2021).

Melihat LSSVM mempunyai kemampuan meningkatkan optimasi dalam klasifikasi dan pentingnya *feature selection* dalam membangun model pengklasifikasi dengan performa lebih baik, maka akan dilakukan penelitian implementasi algoritma LSSVM dengan ReliefF (K=10) *feature selection* (LSSVM-ReliefF) yang diintegrasikan dengan metode SMOTE (LSSVM-ReliefF + SMOTE) untuk mengatasi permasalahan *imbalance class* / ketidakseimbangan kelas pada dataset prediksi cacat perangkat lunak projek NASA MDP Promise. Hasil penelitian ini akan menunjukkan performa dari metode LSSVM-ReliefF + SMOTE dalam mengklasifikasi data prediksi cacat perangkat lunak.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah yang telah diuraikan, maka perumusan masalah dalam penelitian ini adalah bagaimana mengimplementasikan LSSVM-ReliefF dengan SMOTE dalam mengklasifikasi data prediksi cacat perangkat lunak.

1.3 Ruang Lingkup

Ruang lingkup / batasan yang digunakan dalam penelitian ini, yaitu:

1. Pengenalan dilakukan untuk mengklasifikasikan modul *defect* atau *non-defect*.

2. Metode yang digunakan adalah *Least Square Support Vector Machine* (LSSVM) + SMOTE dengan kernel *Linear*, *Polynomial* dan *Radial Basis Function (RBF).*
3. *Feature Selection* yang akan digunakan adalah ReliefF (K=10).
4. *Dataset* yang digunakan adalah *dataset* prediksi cacat perangkat lunak berupa *software metrics* pada projek NASA MDP (*Metrics Data Program*) Promise yang bersifat publik (NASA Metrics Data Program Datasets, 2004), antara lain CM1, KC1, KC2, dan PC1.
5. Pembagian Dataset menggunakan *K-Fold Cross Validation* dengan 10 *fold* (Kulamala et al., 2021)
6. Performa dievaluasi dengan menggunakan *Confusion Matrix*, berupa *precision*, *recall*, dan *accuracy*.
7. Bahasa pemrograman yang digunakan adalah Python.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah mengimplementasikan algoritma LSSVM-ReliefF dengan SMOTE dalam mengklasifikasi data prediksi cacat perangkat lunak (*software defect prediction*) pada kernel *Linear*, *Polynomial* dan *Radial Basis Function*.

1.5 Manfaat Penelitian

Adapun manfaat dari penelitian ini adalah

1. Memahami penerapan metode *Least Square Support Vector Machine* dengan perbandingan pada kernel *Linear*, *Polynomial* dan *Radial Basis Function (RBF).*
2. Memahami penerapan metode SMOTE (*Synthetic Minority Oversampling Technique*) dalam mengatasi *imbalance class*.
3. Memahami penerapan ReliefF (K=10) sebagai *feature selection*.
4. Berkontribusi pada penelitian tentang klasifikasi data prediksi cacat perangkat lunak (*software defect prediction*).

1.6 Metodologi Penelitian

Berikut ini merupakan tahapan-tahapan penelitian yang dilakukan penulis dalam mengklasifikasikan data prediksi cacat perangkat lunak (*software defect prediction*) menggunakan LSSVM (*Least Square Support Vector Machine*).

1. Studi Literatur

Tahapan ini dimulai dengan melakukan pembelajaran literatur berupa jurnal dan buku terkait topik penelitian ini, yaitu perbandingan LSSVM - ReliefF dengan SMOTE pada kernel *Linear, Polynomial* dan *Radial Basis Function (RBF)* dalam klasifikasi data prediksi cacat perangkat lunak.

2. Pengumpulan *Dataset*

Pada tahap ini, dilakukan proses pengumpulan *dataset* berupa *dataset* prediksi cacat perangkat lunak projek NASA MDP Promise yang bersifat public (NASA

Metrics Data Program Datasets, 2004). *Dataset* yang dipakai, yaitu CM1, KC1, KC2, dan PC1.

3. Perancangan

Pada tahapan ini dilakukan perancangan sistem yang dibutuhkan. Diawali dengan *input dataset*, lalu dilakukan normalisasi data untuk mengurangi varian distribusi dari *dataset*. Pengurangan varian tersebut bertujuan untuk mempercepat proses training dan meningkatkan performa hasil. Proses normalisasi yang diterapkan adalah *min-max normalization*. Lalu, data akan dilakukan *feature selection* menggunakan ReliefF (K=10). Proses selanjutnya berupa pembagian *dataset* menggunakan *K-Fold Cross Validation* dengan 10 *fold*. Kemudian, hasil data *training* yang telah diolah akan dilakukan *over-sampling* dengan metode SMOTE untuk mengatasi *imbalance class* pada data dengan tujuan menghasilkan model dengan performa yang lebih baik dan menghindari *overfitting*. Kemudian dilakukan pelatihan LSSVM dengan data *training* yang telah seimbang untuk menghasilkan model / sistem yang dibutuhkan. Setelah itu, dilakukan proses klasifikasi dalam skenario pengujian menggunakan kernel *Linear, Polynomial* dan *Radial Basis Function (RBF)*.

4. Implementasi

Pada tahapan ini dilakukan implementasi dari sistem yang telah dirancang sebelumnya agar sistem dapat mengenali dan melakukan klasifikasi terhadap data *training* yang telah diolah sebelumnya.

5. Pengujian

Pada tahapan ini, sistem yang telah dibuat sebelumnya akan melakukan uji coba terhadap data uji. Setelah tahapan uji coba, hasil pengujian dihitung untuk mendapatkan tingkat keberhasilan metode yang digunakan dengan *Confusion Matrix* dalam menghitung nilai *precision*, *recall*, dan *accuracy*.

1.7 Sistematika Penulisan

Sistematika penulisan dari laporan skripsi yang akan dilakukan ini, antara lain.

BAB 1 PENDAHULUAN

Bab ini menjelaskan tentang latar belakang mengapa penelitian ini dilakukan, perumusan masalah yang akan diselesaikan, ruang lingkup pengerjaan, tujuan dan manfaat yang akan diperoleh, metodologi penelitian, serta sistematika penulisan laporan penelitian.

BAB 2 LANDASAN TEORI

Bab ini menjelaskan tentang teori-teori yang berkaitan dengan penelitian klasifikasi prediksi cacat perangkat lunak, *imbalance class*, algoritma LSSVM, metode SMOTE, ReliefF *Feature Selection*, bahasa pemrograman dan *platform* yang digunakan serta penelitian-penelitian terdahulu.

BAB 3 METODE PENELITIAN

Bab ini menjelaskan tentang kebutuhan perangkat keras dan lunak, serta penjelasan tahapan metode penelitian yang digunakan.

BAB 4 HASIL DAN PEMBAHASAN

Bab ini membahas tentang hasil dari pengujian dan analisis dari setiap skenario pengujian yang dilakukan.

BAB 5 PENUTUP

Bab ini berisi penjelasan mengenai kesimpulan dari hasil penelitian yang telah dilakukan, yaitu implementasi LSSVM dalam klasifikasi data prediksi cacat perangkat lunak dengan *feature selection*. Bagian ini juga dilengkapi dengan saran yang bermanfaat bagi pengembangan penelitian selanjutnya.


UNIVERSITAS
MDP

DAFTAR PUSTAKA

Akbar, M. S., & Rochimah, S. (2017). Prediksi Cacat Perangkat Lunak Dengan Optimasi Naive Bayes Menggunakan Gain Ratio. *Jurnal Sistem Dan Informatika*, 147–155. http://repository.its.ac.id/2527/

Aldraimli, M., Soria, D., Parkinson, J., Thomas, E. L., Bell, J. D., Dwek, M. V., & Chaussalet, T. J. (2020). Machine learning prediction of susceptibility to visceral fat associated diseases. *Health and Technology*, *10*(4), 925–944. https://doi.org/10.1007/s12553-020-00446-1

Alsawalqah, H., Faris, H., Aljarah, I., Alnemer, L., & Alhindawi, N. (2017). *Hybrid SMOTE-Ensemble Approach for Software Defect Prediction*. *1*. https://doi.org/10.1007/978-3-319-57141-6

Arifiyanti, A. A., & Wahyuni, E. D. (2020). Smote: Metode Penyeimbang Kelas Pada Klasifikasi Data Mining. *SCAN - Jurnal Teknologi Informasi Dan Komunikasi*, *15*(1), 34–39. https://doi.org/10.33005/scan.v15i1.1850

Bahaweres, R. B., Agustian, F., Hermadi, I., Suroso, A. I., & Arkeman, Y. (2020). Software defect prediction using neural network based smote. *International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, *2020-Octob*(June 2021), 71–76. https://doi.org/10.23919/EECSI50503.2020.9251874

Balogun, A. O., Basri, S., Capretz, L. F., Mahamad, S., Imam, A. A., Almomani, M. A., Adeyemo, V. E., Alazzawi, A. K., Bajeh, A. O., & Kumar, G. (2021). *SS symmetry Based on Dynamic Re-Ranking Strategy*. 1–23.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, *16*(Sept. 28), 321–357. https://doi.org/10.1613/jair.953

Douzas, G., Bacao, F., & Last, F. (2018). Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE. *Information Sciences*, *465*, 1–20. https://doi.org/10.1016/j.ins.2018.06.056

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*(8), 861–874. https://doi.org/10.1016/j.patrec.2005.10.010

Ghotra, B., McIntosh, S., & Hassan, A. E. (2017). A large-scale study of the impact of feature selection techniques on defect classification models. *IEEE International Working Conference on Mining Software Repositories*, 146–157. https://doi.org/10.1109/MSR.2017.18

Iqbal, A., Aftab, S., Ali, U., Nawaz, Z., Sana, L., Ahmad, M., & Husen, A. (2019). Performance analysis of machine learning techniques on software defect prediction using NASA datasets. *International Journal of Advanced Computer Science and Applications*, *10*(5), 300–308. https://doi.org/10.14569/ijacsa.2019.0100538

Johnson, J. M., & Khoshgoftaar, T. M. (2019). Survey on deep learning with class imbalance. *Journal of Big Data*, *6*(1).https://doi.org/10.1186/s40537-019-0192-5

Khadijah, Adorada, A., Wirawan, P. W., & Kurniawan, K. (2020). The Comparison of Feature Selection Methods in Software Defect Prediction. *ICICoS 2020 - Proceeding: 4th International Conference on Informatics and Computational Sciences*, 0–5. https://doi.org/10.1109/ICICoS51170.2020.9299022

Khaulasari, H. (2018). Combine Sampling Least Square Support Vector Machine Untuk Klasifikasi Multi Class Imbalanced Data. *Jurnal Widyaloka IKIP Widya Darma*, *5*(3), 261–278.

Kilicarslan, S., Adem, K., & Celik, M. (2020). Diagnosis and classification of cancer using hybrid model based on ReliefF and convolutional neural network. *Medical Hypotheses*, *137*(December 2019), 109577. https://doi.org/10.1016/j.mehy.2020.109577

Kulamala, V. K., Kumar, L., & Mohapatra, D. P. (2021). Software Fault Prediction Using LSSVM with Different Kernel Functions. *Arabian Journal for Science and Engineering*, *46*(9), 8655–8664. https://doi.org/10.1007/s13369-021-05643-2

Kumar, L., Satapathy, S. M., & Krishna, A. (2018). Application of SMOTE and LSSVM with various kernels for predicting refactoring at method level. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notesfile:///C:/Users/Thingkilia/Downloads/Documents/10237-26419-1-PB.pdf in Bioinformatics): Vol. 11305 LNCS*. Springer International Publishing. https://doi.org/10.1007/978-3-030-04221-9_14

Kumar, L., Sripada, S. K., Sureka, A., & Rath, S. K. (2018). Effective fault prediction model developed using Least Square Support Vector Machine (LSSVM).

Journal of Systems and Software, *137*, 686–712. https://doi.org/10.1016/j.jss.2017.04.016

Kumar, L., & Sureka, A. (2017). Application of LSSVM and SMOTE on Seven Open Source Projects for Predicting Refactoring at Class Level. *Proceedings - Asia-Pacific Software Engineering Conference, APSEC, 2017-Decem*, 90–99. https://doi.org/10.1109/APSEC.2017.15

Leong, W. C., Bahadori, A., Zhang, J., & Ahmad, Z. (2019). Prediction of water quality index (WQI) using support vector machine (SVM) and least square-support vector machine (LS-SVM). *International Journal of River Basin Management*, *19*(2), 149–156. https://doi.org/10.1080/15715124.2019.1628030

NASA Metrics Data Program Datasets. (2004). *Software Defect Datasets*. PROMISE Software Engineering Repository. https://data.mendeley.com/datasets/923xvkk5mm/1

Pak, C., Wang, T. T., & Su, X. H. (2018). An Empirical Study on Software Defect Prediction Using Over-Sampling by SMOTE. *International Journal of Software Engineering and Knowledge Engineering*, *28*(6), 811–830. https://doi.org/10.1142/S0218194018500237

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html#examples-using-sklearn-metrics-confusion-matrix

Qiao, L., Li, X., Umer, Q., & Guo, P. (2020). Deep learning based software defect prediction. *Neurocomputing*, *385*, 100–110. https://doi.org/10.1016/j.neucom.2019.11.067

Rahmansyah, A., Dewi, O., Andini, P., Hastuti, T., Ningrum, P., & Suryana, M. E. (2018). Membandingkan Pengaruh Feature Selection Terhadap Algoritma Naïve Bayes dan Support Vector Machine. *Seminar Nasional Aplikasi Teknologi Informasi (SNATi)*, 1907–5022.

Robnik, M., & Konenko, I. (2003). Theoretical and empirical analysis of ReliefF and RReliefF. *Machine Learning*, *53*(1--2), 23–69.

Sanjaya, C. B., & Rosadi, M. I. (2018). Klasifikasi Buah Mangga Berdasarkan Tingkat Kematangan Menggunakan Least-Squares Support Vector Machine.

Explore IT : Jurnal Keilmuan Dan Aplikasi Teknik Informatika, *10*(2), 1–8. https://doi.org/10.35891/explorit.v10i2.1255

Sava, J. A. (2022). *Information technology (IT) spending forecast worldwide from 2012 to 2023, by segment(in billion U.S. dollars)*. Statista 2022. https://www.statista.com/statistics/268938/global-it-spending-by-segment/

Son, L. H., Pritam, N., Khari, M., Kumar, R., Phuong, P. T. M., & Thong, P. H. (2019). Empirical study of software defect prediction: A systematic mapping. *Symmetry*, *11*(2). https://doi.org/10.3390/sym11020212

Sreedevi, E., Premalatha, V., Sivakumar, S., & Nayak, S. R. (2019). A comparative study on new classification algorithm using NASA MDP datasets for software defect detection. *Proceedings of the International Conference on Intelligent Sustainable Systems, ICISS 2019*, *Iciss*, 312–317. https://doi.org/10.1109/ISS1.2019.8908096

Suykens, J. A. K., & Vandewalle, J. (1999). Least Squares Support Vector Machine Classifiers. *Neural Processing Letters*, 293–300. https://doi.org/10.1023/A:1018628609742

Triyono, A., Trianto, R. B., & Arum, D. M. P. (2021). Penerapan Least Squares Support Vector Machines (LSSVM) dalam Peramalan Indonesia Composite Index. *Jurnal Informatika Universitas Pamulang*, *6*(1), 210. https://doi.org/10.32493/informatika.v6i1.10237

Urbanowicz, R. J., Olson, R. S., Schmitt, P., Meeker, M., & Moore, J. H. (2018). Benchmarking relief-based feature selection methods for bioinformatics data mining. *Journal of Biomedical Informatics*, *85*(3), 168–188. https://doi.org/10.1016/j.jbi.2018.07.015

Van Gestel, T., Suykens, J. A. K., Baesens, B., Viaene, S., Vanthienen, J., Dedene, G., De Moor, B., & Vandewalle, J. (2004). Benchmarking Least Squares Support Vector Machine Classifiers. *Machine Learning*, *54*(1), 5–32. https://doi.org/10.1023/B:MACH.0000008082.80494.e0

Wei, H., Hu, C., Chen, S., Xue, Y., & Zhang, Q. (2019). Establishing a software defect prediction model via effective dimension reduction. *Information Sciences*, *477*, 399–409. https://doi.org/10.1016/j.ins.2018.10.056

Yao, Z., Song, J., Liu, Y., Zhang, T., & Wang, J. (2019). Research on Cross-version Software Defect Prediction Based on Evolutionary Information. *IOP Conference Series: Materials Science and Engineering*, *563*(5). https://doi.org/10.1088/1757-899X/563/5/052092