Jurnal Teknologi Terpadu

https://journal.nurulfikri.ac.id/index.php/jtt
ISSN : 2477-0043 ISSN ONLINE : 2460-7908

KOMBINASI LINIER TARGET DATA UNTUK REGRESI MULTITARGET MENGGUNAKAN *PRINCIPAL COMPONENT ANALYSIS*

Yonathan Purbo Santosa[1]

[1] Teknik Informatika, Universitas Katolik Soegijapranata
Semarang, Jawa Tengah, Indonesia 50275
yonathansantosa@unika.ac.id

Abstract

Linear regression is a method to predict numbers, a dependent variable (output) based on some independent variables (inputs). The problem with regression is that some data does not fall into linear problems. Based on this problem, RLC was invented to randomly find a correlation between output by projecting the data into the higher dimension. Unfortunately, RLC does not provide ways to inverse the projection, resulting in poor performance results. On top of that, projecting the data into a higher dimension will increase the learning algorithm complexity. Consequently, PCA can solve the problems by projecting the target data into a lower dimension while leaving possibilities for inverse transformation. This research was implemented with the help of the sci-kit-learn library to create and train the regression model and transform the dataset using Python programming language. As a result, for 12 datasets, augmentation using PCA achieved lower error in 7 datasets than RLC, averaging at 0.3270 for augmentation using PCA and 0.4003 for augmentation using RLC.

Keywords: *Dimension reduction, linear regression, multidimension regression, multitarget regression, PCA*

Abstrak

Regresi linier adalah metode untuk memprediksi sebuah nilai (variabel dependen) berdasarkan beberapa *input* (variabel independen). Permasalahan pada regresi linier adalah beberapa data tidak termasuk ke dalam kategori linier. Sebuah metode bernama RLC diciptakan untuk menemukan korelasi antara data output dengan cara memproyeksikan data ke dalam dimensi yang lebih tinggi. Sayangnya, metode RLC tidak dapat diinvers transformasinya. Selain itu, dengan memproyeksikan data ke dimensi yang lebih tinggi akan menambah kompleksitas dari algoritma pembelajaran. Oleh karena itu, PCA akan digunakan untuk memecahkan masalah ini dengan cara memproyeksikan data ke dimensi yang lebih rendah sembari mempertahankan kemampuan untuk melakukan invers proyeksi. Penelitian ini diimplementasikan dengan bantuan *library scikit-learn* untuk membuat model regresi dan transformasi data dengan menggunakan bahasa pemrograman Python. Hasilnya, untuk 12 dataset, metode augmentasi PCA mampu mendapatkan nilai *error* yang lebih rendah dalam 7 dataset dibandingkan dengan RLC dengan rata-rata nilai *error* 0.3270 untuk metode augmentasi PCA dan 0.4003 untuk metode augmentasi RLC.

Kata kunci: PCA, reduksi dimensi, regresi linier, regresi multidimensi, regresi multitarget

1. PENDAHULUAN

Pada analisis statistik, ada metode yang dapat digunakan untuk memprediksi suatu variabel dependen (*output)* berdasarkan beberapa data yang diberikan sebagai variabel independen (*input*) yang dikenal sebagai metode regresi. Regresi mencari dan menghitung korelasi antara *output* berdasarkan semua variabel *input* untuk mendefinisikan suatu fungsi yang akan memetakan *input* ke *output* [1]. Kita hanya dapat melakukan fungsi estimasi yang akan memetakan *input* ke *output* dengan menemukan deviasi minimum antara fungsi dan variabel dependen karena data yang dikumpulkan mungkin memiliki *noise*. Dalam kasus linier, korelasi antara variabel dependen dan parameter fungsi dapat diselesaikan dengan fungsi linier seperti fungsi garis, yang biasanya dapat diselesaikan dengan menggunakan estimasi *least square* [2].

Pada masalah yang lebih kompleks, yang terdiri dari *input* dan *output* multidimensi, sering kali tidak termasuk dalam kategori yang dapat diselesaikan secara linier sehingga tidak dapat diselesaikan dengan menggunakan estimasi *least square*. Terlebih, metode regresi hanya mampu

Diterima 14 Juni 2022, Direvisi 10 Desember 2022, Diterima untuk publikasi 16 Desember 2022

menghasilkan satu nilai prediksi sehingga untuk kasus variabel dependen multidimensi diperlukan beberapa model regresi. Padahal banyak sekali kasus dalam dunia industri yang memerlukan metode prediksi untuk variabel dependen dengan multidimensi [3], [4], terutama dimasa industri 4.0 yang menyebabkan data dengan dimensi yang banyak adalah hal yang sangat wajar [5]–[7]. Selain itu pada bidang kesehatan, model regresi juga terus menjadi kebutuhan utama dalam kasus prediksi suatu penyakit [8]. Didorong oleh kebutuhan untuk memecahkan masalah regresi non-linier dan multidimensi di dunia industri, peneliti di seluruh dunia untuk berusaha untuk menggunakan metode penyelesaian data multidimensi yang non-linier dengan model *machine learning* yang lebih kuat, seperti jaringan syaraf tiruan [9], *deep regression* [10], [11], *random linear target combination* [12], *variational autoencoder regression* [13], *support vector regression*[14], dan lainnya.

Tingginya kebutuhan untuk melakukan komputasi yang lebih kompleks, kemampuan komputer dalam memproses data juga harus ikut meningkat, terutama dalam permasalahan *machine learning* [15]. Tak lain halnya kebutuhan untuk memecahkan permasalahan regresi multidimensi secara akurat pun ikut bertambah seiring dengan kebutuhan industri terutama dalam kasus *big data regression* [16]. Padahal semakin besar dimensi sebuah data, akan semakin besar pula probabilitas untuk terjadi kesalahan dalam pengukuran dan pelabelan pada data [17]. Berdasarkan permasalahan di atas, metode untuk membuat data lebih mudah dipelajari banyak penelitian yang cenderung melakukan pemrosesan pada data ketimbang pada algoritma pembelajaran.

Untuk menyelesaikan permasalahan regresi multidimensi, metode regresi yang sering digunakan adalah dengan menggunakan regresi *single target* (ST) untuk kemudian digabung menjadi sebuah model yang lebih kompleks menggunakan metode *ensemble* seperti pada penelitian Boye dkk. [18] untuk melakukan estimasi harga unit rumah dan berhasil memperoleh nilai standar *error residual* 0.027 satuan. Herawati dkk. [19] membandingkan beberapa metode regresi kembangan untuk menyelesaikan masalah multi-kolinier dalam data dengan cara mengurangi koliniaritas dari variabel independen. Hasilnya menunjukkan *principal component regession* adalah metode dengan nilai *average mean squared error* terendah yaitu 0.0014 dibanding *ridge regression* yang memperoleh nilai *error* 0.4099 [19]. Hal ini menunjukkan dengan mengurangi koliniaritas dari variabel independen, dapat meningkatkan akurasi dari model regresi.

Penelitian yang dilakukan oleh Tsoumakas dkk. [12] memiliki metode pendekatan yang lain, yaitu bahwa koliniaritas dari variabel independen akan berkurang dengan cara melakukan transformasi terhadap variabel dependen. Dalam penelitian tersebut, variabel dependen ditransformasi secara *random* dengan hanya memilih beberapa atribut pada variabel dependen dan melakukan transformasi ke dimensi yang lebih tinggi. Metode tersebut diberi nama *random linear target combination* (RLC) pada regresi ST. Meskipun menggunakan kombinasi *random*, hasil dari penelitian tersebut menunjukkan performansi yang lebih baik (dinilai dari nilai *error*) dibandingkan dengan *state-of-the-art* sebelumnya yaitu *multi-objective random forest algorithm* (MORF) yang hanya lebih unggul dalam 4 *dataset* dari 12 *dataset* yang diuji coba .

Pada metode augmentasi RLC, data target ditransformasi ke dimensi yang lebih tinggi hingga 30 kali lipat lebih besar, sehingga dapat menyebabkan berkurangnya stabilitas dari sebuah model *regressor* yang sering dikenal dengan istilah *curse of dimensionality* di dalam *data mining* maupun *machine learning* [20]. Selain itu semakin tinggi dimensi sebuah data, semakin tinggi pula jumlah *noise* yang terdapat pada data ketika data tersebut dibuat atau dikumpulkan [21]. Hal ini dianggap sebuah permasalahan penting karena dalam proses regresi, keakuratan dari prediksi sangat penting.

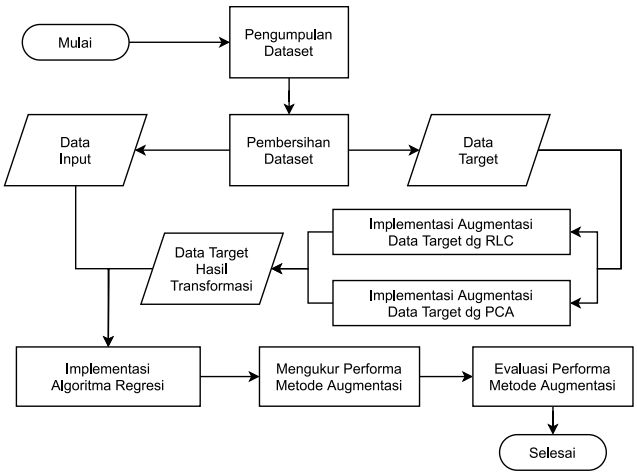
Tidak hanya permasalahan dimensi, proses transformasi data target dengan metode RLC menggunakan perkalian matriks yang tidak dapat dilakukan transformasi inversnya karena matriks transformasi tersebut tidak memenuhi syarat matriks yang dapat dihitung inversnya. Hal ini menyebabkan hasil dari prediksi tidak dapat digunakan untuk menghasilkan nilai prediksi dengan satuan yang sama dengan data. Sebagai contoh, model regresi untuk harga rumah harus bisa menghasilkan prediksi berupa harga rumah [18]. Oleh sebab itu, untuk menghitung matriks invers matriks tersebut harus menggunakan metode estimasi *Moore-Penrose Psuedoinverse* (MPPI). Dikarenakan metode transformasi tersebut adalah sebuah estimasi, keakuratan dari model *regresi* akan tergantung dari keakuratan proses estimasi invers matriks tersebut. Hal ini didukung oleh penelitian Górecki dan Łuczak [22], yang dalam pengujian menggunakan 15 *dataset*, algoritma genetika sebagai metode generalisasi dari *psuedoinverse* lebih unggul dalam 14 *dataset* dibandingkan dengan hanya menggunakan MPPI. Sayangnya, algoritma genetika berjalan sangat lamban untuk mencapai titik optimal jika *dataset* terlalu besar dikarenakan kompleksitas komputasi dari algoritma genetika [22], [23]. Hal ini mendorong untuk mencari metode transformasi data target yang dapat dihitung nilai inversnya tanpa metode estimasi guna memperoleh nilai *error* terendah tanpa mengorbankan komputasi yang kompleks.

Keterbatasan pada penelitian terdahulu dalam hal mengatasi permasalahan *curse of dimensionality* dan penggunaan matriks *psuedoinverse* yang dapat menyebabkan kurang akuratnya hasil prediksi, mendorong penelitian ini untuk dilakukan dengan tujuan memberikan usulan metode pengganti yang bekerja dengan cara mengurangi dimensi melalui kombinasi lainnya guna mencegah *curse of dimensionality* tanpa menggunakan metode estimasi *psuedoinverse* dalam melakukan proses transformasi.

Dengan demikian, hasil dari regresi diduga akan memiliki nilai *error* yang lebih rendah dalam rangka mendukung industri 4.0 dalam melakukan regresi linier terutama untuk data dengan variabel dependen yang memiliki multidimensi.

2. METODE PENELITIAN

Pada bagian menjelaskan metode untuk melakukan analisa terhadap efek dari perubahan dimensi pada data target, baik dari penelitian sebelumnya yaitu RLC dan metode usulan yang digunakan pada penelitian ini. Alur penelitian ini dirancang seperti pada Gambar 1 yang akan dijabarkan secara lebih mendalam pada bagian berikutnya.



Gambar 1. Diagram Alir Penelitian

Uji coba dan evaluasi perbandingan metode transformasi data serta implementasi model regresi linier dibuat menggunakan bahasa pemrograman Python versi 3.7 dengan bantuan *library machine learning* Scikit-learn versi 1.1.3 [24], dan Numpy versi 1.23 [25]. *Library* Scikit-learn memuat *Gradient Tree Boosting* untuk melakukan ensambel regresi linier serta transformasi data yang dibantu oleh *library* Numpy. Program diimplementasikan menggunakan komputer dengan spesifikasi Windows 10, Intel Core I7-6200U, RAM 16GB, dan *harddisk* 500GB.

2.1 Pengumpulan *Dataset*

Pengujian metode transformasi membutuhkan *dataset* yang digunakan sebagai bahan percobaan dalam memperoleh hasil pengukuran metode augmentasi yang digunakan. Jika mengacu kepada penelitian yang dilakukan Tsoumakas dkk. [12], setidaknya 12 *dataset* yang sama juga digunakan pada penelitian ini seperti yang dijabarkan pada Tabel 1. Hal ini bertujuan untuk mendapatkan komparasi yang sama jika dibandingkan dengan penelitian terdahulu [12]. Deskripsi lengkap serta metode pengumpulan data dari masing-masing *dataset* dijabarkan secara lengkap pada penelitian yang menyediakan *dataset* tersebut [12], [26]–[29]. *Dataset* dapat didapatkan pada sumber penyedia *dataset* dan dapat diunduh secara bebas. Pada Tabel 1, dijabarkan bahwa masing-masing *dataset* memiliki data target dengan ukuran dimensi q. *Dataset* tersebut kemudian diunduh dan diproses lebih lanjut menggunakan bahasa pemrograman Python dan *library* Numpy agar dapat dilakukan transformasi data dan untuk melakukan proses regresi linier melalui program. Proses transformasi data target dari masing-masing *dataset* dilakukan menggunakan metode PCA dan RLC yang akan digunakan untuk menguji keefektifan kedua algoritma tersebut dalam menangani permasalahan regresi multitarget.

Tabel 1. *Dataset* dengan Deskripsi

Nama	Singkatan	$\|D\|$	q
Airline Ticket Price 1 [28]	atp1d	337	6
Airline Ticket Price 2 [28]	atp7d	296	6
Occupational Employment Survey 1 [28]	oes10	334	16
Occupational Employment Survey 2 [28]	oes97	403	16
River Flow 1 [28]	rf1	4165/5065	8
River Flow 2 [28]	rf2	4165/5065	8
Supply Chain Management 1 [28]	sf1978	8145/1658	16
Supply Chain Management 2 [28]	sf1969	7463/1503	16
Electrical Discharge Machining [29]	edm	154	2
Solar Flare 1 [26]	scm1d	323	3
Solar Flare 2 [26]	scm20d	1066	3
Water Quality [27]	wq	1060	14

Penjelasan singkat mengenai *dataset* pada Tabel 1 akan dijelaskan pada bagian di bawah. *Dataset Airline Ticket Price* adalah *dataset* yang berisi harga tiket pesawat untuk 1 hari ke depan (atp1d) dan harga tiket terendah untuk 7 hari ke depan (atp7d) untuk maskapai Delta, maskapai Continental, maskapai Airtran, dan maskapai United [28]. *Dataset Occupational Employment Survey* terdiri dari data penerimaan sebagai karyawan yang dilakukan oleh Badan Statistik Kepegawaian Amerika Serikat (*US Bureau of Labor Statistics*) pada tahun 2010 (oes10) dan pada tahun 1997 (oes97) [28]. *Dataset River Flow* dibuat dalam rangka melakukan prediksi terhadap jaringan sungai pada 8 kota yang berbeda untuk 48 jam ke depan [28]. *Dataset River Flow* ada 2 jenis data, *dataset* pertama berisi observasi aliran (rf1) sedangkan *dataset* kedua ditambahkan juga data mengenai kelembapan udara [28]. *Dataset Supply Chain Management* didapatkan dari *Trading Agent Competition in Supply Chain Management* (TAC SCM) pada tahun 2010 [28]. *Dataset* tersebut terdiri dari data dengan prediksi untuk 1 hari ke depan (scm1d) dan data dengan prediksi untuk 20 hari ke depan (scm20d) [28]. *Dataset Electrical Discharge Machining* berisi data yang mereproduksi perilaku manusia yang bertugas sebagai petugas dalam mengoperasikan sebuah mesin [29]. *Dataset Solar Flare* berisi tentang tipe *solar flare* yang diamati selama kurun waktu 24 jam [26]. Terdapat dua jenis data, pertama data pengamatan pada tahun 1969 dan kedua tahun 1978 [26]. *Dataset Water Quality* berisi tentang kondisi fisik dan kimia pada kualitas air di sungai Slovenia [27]. Semua *dataset* tersebut diunduh dalam format *csv* untuk dapat dibaca oleh *library* Numpy menggunakan bahasa pemrograman Python.

2.2 Pembersihan *Dataset*

Data yang sudah diunduh harus diproses terlebih dahulu untuk membersihkan data yang tidak lengkap dan data yang tidak dapat diolah oleh regresi seperti data yang berupa teks dan kategorial. Data yang termasuk ke dalam kategori tersebut di atas, diubah ke dalam bentuk angka yang merepresentasikan tipe data yang ada pada masing-masing *dataset*. Selain itu dikarenakan data mengandung satuan nilai yang berbeda-beda, data juga akan dilakukan pemrosesan awal *MinMaxScaler* seperti yang dijabarkan pada persamaan (1) dan (2). Hal ini bertujuan untuk mempermudah proses perbandingan dari kedua metode augmentasi PCA dan RLC agar dapat dilakukan dengan mudah.

$$X_\sigma = \frac{x - min(X)}{max(X) - min(X)} \quad (1)$$

$$\hat{x} = X_\sigma * (max(X) - min(X)) + min(X) \quad (2)$$

2.3 Implementasi Augmentasi Data Target menggunakan RLC

Dalam melakukan augmentasi data target dua metode akan digunakan dalam penelitian ini. Metode yang pertama adalah metode RLC yang menjadi landasan dalam penelitian ini dan metode PCA sebagai alternatif yang akan menggantikan RLC. Metode RLC sendiri berusaha menemukan korelasi linier antar data target dengan cara melakukan inisialisasi sebuah matriks kombinasi *C* secara *random* dengan tahapan sebagai berikut:

1. Pilih ukuran dimensi data target yang baru r. Pada penelitian sebelumnya, parameter r yang digunakan adalah 500 [12].
2. Pilih jumlah data target dari data asli yang akan dikorelasikan k . Pada penelitian sebelumnya, parameter k berkisar antara 2 hingga q.
3. Buat matriks kosong C dengan ukuran $q \times r$
4. Untuk masing-masing kolom pada C, pilih k buah baris dan isikan nilai dengan nilai *random* yang diambil dari distribusi uniform dengan rentang 0 hingga 1 inklusif.

Berdasarkan langkah-langkah di atas dapat diilustrasikan pada Gambar 2. Sebagai contoh terdapat 3 buah baris data dengan 2 data target Y yang diilustrasikan pada Gambar 2(a). Kemudian dibuatlah sebuah matriks C yang diisi dengan nilai random, diilustrasikan dengan Gambar 2(b). Hasil perkalian antara data target Y dengan matriks C kemudian akan digunakan untuk melakukan proses pelatihan model regresi linier yang diilustrasikan pada Gambar 2(c). Proses kombinasi target yang dijabarkan pada dilakukan hanya untuk *training* data, sehingga akan dihasilkan sebuah matriks baru $\hat{Y}$ yang akan digunakan sebagai data target dalam melakukan proses regresi menggunakan data target $\hat{Y}$ seperti pada Gambar 2(c). Tsoumakas dkk. [12] tidak melakukan proses invers untuk mengubah $\hat{Y}$ menjadi Y kembali.

1	2
4	1
3	4

(a)

0.2	0	0	0.5	0.3	0.3
0.1	0.4	0.1	0	0.2	0

(b)

0.4	0.8	0.2	0.5	0.9	0.3
0.9	0.4	0.1	2	1.4	1.2
1	1.6	0.4	1.5	1.7	0.9

(c)

Gambar 2. (a) Data target asli Y, (b) Matrix Kombinasi C, (c) Data target dari hasil kombinasi.

Pada penelitian ini akan dilakukan proses invers menggunakan MPPI terhadap matriks C untuk menemukan C^+. Asumsi tersebut didasarkan bahwa ada sebuah matriks C^+ yang memenuhi persamaan berikut:

$$C^+ = (C^\top C)^{-1} C^\top \quad (3)$$

$$C^+ \approx C^{-1} \quad (4)$$

Dari persamaan (3), dapat dihitung kembali transformasi inversnya agar dapat dilihat nilai prediksi yang mengacu kepada *dataset*. Proses invers transformasi tersebut dijabarkan pada persamaan (5).

$$Y = \hat{Y} \times C^+ \quad (5)$$

Semua proses yang telah dijabarkan diimplementasikan menggunakan bahasa pemrograman Python dan *library Numpy* dan *Scikit-learn*. Setelah diimplementasikan, program dapat digunakan untuk melakukan augmentasi pada data yang sudah dibersihkan.

2.4 Implementasi Augmentasi Data Target menggunakan PCA

Metode kedua yang merupakan metode usulan dari penelitian ini menggunakan metode transformasi PCA. Proses transformasi PCA membutuhkan perhitungan matriks C yang didapatkan dari *singular value decomposition* (SVD). Proses SVD menghasilkan *principal components* (PCs) yang digunakan untuk mencari data yang dengan jumlah variasi tertinggi. Dengan melakukan limitasi terhadap sebagian data yang memiliki variasi tertinggi, data dapat diproyeksikan ke dimensi yang lebih rendah dengan langkah-langkah sebagai berikut:

1. Pilih ukuran dimensi data target yang baru k, yang berada pada rentang antara 2 hingga q
2. Hitung menggunakan SVD untuk mendapatkan *eigenvector* C dan *eigenvalue* Σ
3. Pilih sub set dari Σ dengan ukuran k secara diagonal untuk membuat matriks ukuran urut mulai dari PCs yang paling signifikan
4. Lakukan transformasi berdasarkan matriks C

Persamaan (6) adalah persamaan *singular value decomposition* (SVD) untuk matriks kovarian S dari data target Y dan dapat dihitung matriks dekomposisinya yang merupakan matriks transformasi C menggunakan SVD. Matriks C pada persamaan (6) hingga (8) adalah matriks ortogonal yang berisi eigenvector dari matriks kovarian S, Σ adalah matriks diagonal yang berisikan *singular value* dari S, $\hat{Y}$ adalah hasil transformasi dari matriks Y menggunakan matriks C.

$$Y^\top Y = S = U\Sigma C^\top \quad (6)$$

$$\hat{Y} = Y \times C \quad (7)$$

$$Y = \hat{Y} \times C^\top \quad (8)$$

Dikarenakan matriks C adalah matriks ortogonal, invers dari matriks C sama dengan *transpose* dari matriks C. Dengan demikian, proses transformasi dapat memenuhi sistem persamaan linier (7) dan (8). Hal ini menunjukkan proses transformasi dapat dicari transformasi inversnya tanpa metode estimasi seperti MPPI.

2.5 Implementasi Model Regresi

Pengujian metode augmentasi data dilakukan dengan model regresi linear yang melakukan regresi linier terhadap masing-masing variabel target (ST singkatan dari satu target) dengan menggunakan algoritma *decision tree regressor* yang dilatih menggunakan metode ensambel *gradient boosting.* Untuk mengakomodasi seluruh variabel target, maka akan diperlukan sebanyak k buah ST untuk masing-masing metode augmentasi untuk setiap dataset. Parameter dari model regresi yang digunakan mengacu pada parameter yang didefinisikan dalam penelitian sebelumnya [12]. Model regresi diimplementasikan menggunakan bahasa pemrograman Python dengan *library Scikit-learn*. Selain parameter yang dijabarkan di dalam Tabel 2, semua parameter menggunakan nilai *default* dari implementasi *library Scikit-learn* [24]. Selain itu, untuk parameter *random generator* akan menggunakan angka 0.

Tabel 2. Parameter *Gradient Boosting Tree Regressor*

Parameter	Nilai
Jumlah maksimum *leaf node*	4
Jumlah *decision tree regressor*	100

2.6 Pengukuran Performa Metode Augmentasi

Dalam rangka mengetahui performa dari masing-masing metode, *cost function Average Relative Root Mean Squared Error* (aRRMSE) yang telah didefinisikan di dalam penelitian yang menjadi landasan penelitian ini oleh Tsoumakas dkk. [12] akan digunakan sebagai alat ukur. Sebelum perhitungan aRRMSE dilakukan, proses invers dari augmentasi RLC dan PCA, serta *MinMaxScaler* akan dilakukan. aRRMSE sendiri dijabarkan dalam persamaan (9) dan (10).

$$RRMSE = \sqrt{\frac{\sum_{(x,y)\in D_{test}} \left(h(\mathrm{x})_j - y_j\right)^2}{\sum_{(x,y)\in D_{test}} \left(\bar{y}_j - y_j\right)^2}} \quad (9)$$

$$aRRMSE(h, D_{test}) = \frac{1}{k}\sum_{j=1}^{k} RRMSE \quad (10)$$

Pada persamaan (9), $h(\mathrm{x})_j$ adalah *output* dari regresi linier untuk seluruh data *input* x untuk data target y_j sedangkan $\bar{y}_j$ nilai rata-rata untuk data target ke-j, k adalah jumlah dimensi dari data target. Dengan kata lain, fungsi aRRMSE, masing-masing output dari regresi linier akan dihitung nilai deviasinya terhadap data target dengan mengacu pada nilai rata-rata masing-masing data target. Hal ini bertujuan untuk melakukan standarisasi performa melalui rata-rata dari data target yang tidak homogen setelah proses *training* model ensambel [12].

2.7 Evaluasi Performa Kedua Metode Augmentasi

Di akhir dari proses pelatihan model selesai, hasil dari perhitungan nilai *error* menggunakan *cost function* dari kedua belas *dataset* terhadap data tes akan dibandingkan dengan cara mencari nilai terendah untuk masing-masing *dataset*. Metode dengan nilai *error* yang lebih rendah adalah metode yang dianggap lebih berhasil dalam melakukan regresi linier. Jumlahan keberhasilan dari masing-masing metode untuk semua *dataset* kemudian dibandingkan untuk memperoleh kesimpulan.

3. HASIL DAN PEMBAHASAN

3.1 Hasil Uji Coba Metode *Random* Linear Target *Combination*

Hasil uji coba metode RLC dijabarkan dalam Tabel 3. Nilai k pada Tabel 3 merupakan jumlahan data target yang dikorelasikan secara linier dan *random*. Pada metode RLC dimensi dari data target akan menjadi konstan yaitu 500 dimensi. Hasil uji coba ini akan menjadi tolok ukur dari uji coba metode PCA. Untuk mempermudah interpretasi dari hasil uji coba, nilai *error* yang merupakan nilai minimum untuk masing-masing *dataset* dicetak tebal dan digarisbawah.

Mengacu kepada hasil uji coba yang dilakukan oleh Tsoumakas dkk. [12] pada penelitian sebelumnya dengan *library* yang berbeda pada Tabel 4, jika dibandingkan dengan Tabel 3 dapat terlihat bahwa hasil uji coba yang dilakukan memiliki nilai *error* yang lebih rendah. Perbedaan ini diduga dikarenakan perbedaan *library* dan *tools* yang digunakan yang diduga menyebabkan proses *random* yang berbeda. Oleh karena hasil dari implementasi di dalam penelitian ini memiliki rata-rata nilai *error* yang lebih rendah, maka hasil uji coba dengan metode RLC pada penelitian ini akan digunakan sebagai acuan.

Tabel 3. Hasil Uji Coba Metode RLC

k	atp1d	atp7d	edm	oes10	oes97	rf1	rf2	sf1978	sf1969	wq	scm1d	scm20d
2	0.4137	0.4078	**0.6703**	**0.5649**	**0.5247**	0.1710	0.1830	1.1424	**0.5188**	0.1021	0.1140	0.0998
3	0.4016	0.3794		0.5726	0.5255	0.1687	0.1802	**1.1268**	0.5377	0.0982	**0.1139**	0.0986
4	0.4032	0.3671		0.5789	0.5351	0.1656	0.1784			0.0938	0.1168	**0.0973**
5	0.3965	**0.3659**		0.5907	0.5284	0.1665	0.1790			**0.0934**	0.1171	0.0977
6	**0.3900**	0.3660		0.5822	0.5345	**0.1638**	0.1739			0.0956	0.1194	0.0977
7				0.5876	0.5408	0.1666	0.1741			0.0966	0.1199	0.0989
8				0.5961	0.5385	0.1654	**0.1738**			0.0950	0.1215	0.0986
9				0.5871	0.5415					0.1009	0.1226	0.0988
10				0.5876	0.5413					0.1006	0.1221	0.0986
11				0.5980	0.5413					0.1032		0.1007
12				0.6029	0.5414					0.1049		0.1000
13				0.5912	0.5404					0.1109		0.1003
14				0.5957	0.5401					0.1172		0.1011
15				0.5985	0.5438							0.1012
16				0.6038	0.5424							0.1011
MIN	0.3900	0.3659	0.6703	0.5649	0.5247	0.1638	0.1738	1.1268	0.5188	0.0934	0.1139	0.0973

Tabel 4. Hasil Uji Coba RLC dari Penelitian Sebelumnya oleh Tsoumakas dkk. [12]

k	atp1d	atp7d	edm	oes10	oes97	rf1	rf2	sf1978	sf1969	wq	scm1d	scm20d
2	0.3842	**0.4614**	**0.6996**	**0.5026**	0.5593	**0.7265**	**0.7036**	1.2312	1.5746	0.9100	**0.4572**	0.7469
3	**0.3840**	0.4653		0.5084	**0.5588**	0.7878	0.7584	**1.2172**	**1.5675**	**0.9080**	0.4610	**0.7467**
4	0.3884	0.4796		0.5232	0.5730	0.8204	0.7922			0.9085	0.4663	0.7472
5	0.3952	0.4917		0.5359	0.5837	0.8584	0.8327			0.9086	0.4699	0.7477
6	0.4022	0.5029		0.5472	0.5889	0.8515	0.8257			0.9089	0.4775	0.7490
7				0.5551	0.5958	0.8446	0.8106			0.9090	0.4820	0.7513
8				0.5734	0.6076	0.8868	0.8655			0.9107	0.4855	0.7536
9				0.5911	0.6153					0.9122	0.4889	0.7548
10				0.6031	0.6229					0.9128	0.4932	0.7537
11				0.6154	0.6348					0.9150	0.4978	0.7573
12				0.6285	0.6449					0.9163	0.5020	0.7571
13				0.6354	0.6590					0.9188	0.5057	0.7619
14				0.6428	0.6682					0.9217	0.5133	0.7640
15				0.6525	0.6860						0.5155	0.7681
16				0.6652	0.6916						0.5218	0.7704
MIN	0.3840	0.4614	0.6996	0.5026	0.5588	0.7268	0.7036	1.2172	1.5675	0.9080	0.4572	0.7467

3.2 Hasil Uji Coba Metode *Principal Component Analysis*

Hasil uji coba metode PCA disajikan pada Tabel 5, masing-masing *dataset* memiliki jumlah uji coba yang berbeda karena perbedaan jumlah data target seperti yang dijabarkan pada Tabel 1. Tergantung dari jumlah data targetnya (q), untuk masing-masing proyeksi ke dimensi yang lebih rendah (k) dari data target akan digunakan untuk melatih *decision tree regressor* dengan *gradient boosting*. Sebagai contoh, untuk *dataset* atp1d yang memiliki $q = 6$, akan dilakukan uji coba dengan memproyeksikan data target dengan dimensi $k \in \{2, 3, 4, 5, 6\}$. Dari hasil pengujian tersebut akan diambil parameter k mana yang memiliki nilai *error* paling rendah untuk tiap-tiap *dataset.* Dapat diamati juga pada Tabel 5 pola dari masing-masing *dataset* yang memiliki nilai *error* yang paling rendah, yaitu pada nilai k yang cenderung lebih kecil (berkisar antara 2 hingga 5), sehingga semakin besar dimensi yang dimiliki oleh data target dari masing-masing dataset, semakin besar pula jumlah reduksi dimensi yang terjadi untuk mendapatkan nilai *error* yang lebih rendah pada pelatihan model regresi ketimbang metode RLC yang mentransformasi data targetnya ke dalam 500 dimensi.

Tabel 5. Hasil Uji Coba PCA

k	atp1d	atp7d	edm	oes10	oes97	rf1	rf2	sf1978	sf1969	wq	scm1d	scm20d
2	0.3921	**0.3755**	**0.8731**	**0.3849**	0.3934	0.1801	0.1895	0.5368	**0.3372**	**0.0804**	**0.1196**	**0.0870**
3	**0.3855**	0.4681		0.4521	**0.3842**	0.1807	0.1898	**0.5362**	0.3907	0.0836	0.1197	0.0872
4	0.3870	0.5049		0.4495	0.4018	0.1741	0.1905			0.0846	0.1198	0.0876
5	0.3890	0.5110		0.4509	0.4137	**0.1722**	**0.1878**			0.0855	0.1224	0.0916
6	0.3870	0.5115		0.4486	0.4460	0.1756	0.1900			0.0888	0.1267	0.0959
7				0.4489	0.4522	0.1759	0.1903			0.0929	0.1275	0.0983
8				0.4514	0.4514	0.1759	0.1903			0.0918	0.1356	0.1191
9				0.4442	0.4504					0.0947	0.1347	0.1165
10				0.4444	0.4499					0.0949	0.1360	0.1199
11				0.4449	0.4515					0.0996	0.1393	0.1223
12				0.4451	0.4500					0.0987	0.1397	0.1217
13				0.4449	0.4504					0.0988	0.1397	0.1232
14				0.4457	0.4512					0.0999	0.1434	0.1258
15				0.4455	0.4512						0.1457	0.1275
16				0.4461	0.4512						0.1483	0.1279
MIN	0.3855	0.3755	0.8731	0.3849	0.3842	0.1722	0.1878	0.5362	0.3372	0.0804	0.1196	0.0870

Pada Tabel 6, hasil uji coba untuk masing-masing *dataset* diurutkan mulai dari *dataset* yang memiliki dimensi data target yang paling kecil hingga *dataset* yang memiliki dimensi data target yang paling besar. Dapat terlihat dari hasil uji coba tersebut, persentase reduksi dimensi yang dibutuhkan untuk memperoleh nilai *error* yang kecil. Untuk *dataset* dengan jumlah dimensi yang lebih dari 10, nilai *error* minimum dapat dicapai menggunakan reduksi hingga 88% dari dimensi aslinya.

Tabel 6. Persentase Reduksi Dimensi terhadap Nilai *Error* Minimum

Dataset	k	q	Persentase Reduksi Dimensi
edm	2	2	0%
sf1969	2	3	33%
sf1978	3	3	0%
atp1d	3	6	50%
atp7d	2	6	67%
rf1	5	8	38%
rf2	5	8	38%
wq	2	14	86%
scm1d	2	16	88%
oes97	3	16	81%
scm20d	2	16	88%
oes10	2	16	88%

3.3 Perbandingan RLC dan PCA dalam Regresi Multitarget

Hasil dari uji coba RLC dan PCA dibandingkan dengan cara membandingkan tiap-tiap *dataset* dengan hasil *error* yang didapatkan, hanya hasil *error* terkecil yang akan dibandingkan. Rangkuman hasil *error* tersebut dapat dijabarkan di dalam Tabel 7 dengan nilai rata-rata keseluruhan uji coba terhadap dua belas *dataset* disajikan pada baris terakhir.

Tabel 7. Perbandingan aRRMSE metode RLC dan PCA

Dataset	RLC	PCA
atp1d	0.39	**0.3855**
atp7d	**0.3659**	0.3755
edm	**0.6703**	0.8731
oes10	0.5649	**0.3849**
oes97	0.5247	**0.3842**
rf1	**0.1638**	0.1722
rf2	**0.1738**	0.1878
sf1978	1.1268	**0.5362**
sf1969	0.5188	**0.3372**
wq	0.0934	**0.0804**
scm1d	**0.1139**	0.1196
scm20d	0.0973	**0.087**
AVG	**0.4003**	**0.327**

Dari hasil uji coba tersebut dapat dilihat bahwa metode PCA memiliki nilai *error* yang lebih rendah untuk *dataset* atp1d, oes10, oes97, sf1978, sf1969, wq, dan scm20d. Sehingga jika dihitung jumlah *dataset* berdasarkan metode yang memiliki *error* yang lebih rendah, didapatlah perbandingan RLC : PCA = 5 : 7. Selain perbandingan untuk masing-masing *dataset*, metode RLC dan PCA memiliki hasil rata-rata *error* masing-masing untuk semua *dataset* yaitu 0.4003 dan 0.3270. Dari hasil perbandingan *error* kedua metode tersebut, dapat ditentukan bahwa PCA mampu menghasilkan nilai *error* yang lebih rendah untuk

sebagian besar dari *dataset* yang diuji coba pada penelitian ini sebagai kesimpulan

3.4 Implementasi Metode PCA pada Penelitian Lanjutan

Pada hasil penelitian ini, dapat dibuktikan bahwa transformasi menggunakan metode PCA dapat menghasilkan nilai *error* yang lebih rendah dibandingkan dengan metode RLC. Selain itu, PCA mampu mereduksi jumlah dimensi dari data target. Hal ini tentu dapat membantu mengurangi nilai *error* dalam membuat sebuah model regresi. Dari temuan tersebut, metode PCA dapat digunakan sebagai sebuah metode pemrosesan awal dalam membuat sebuah model regresi, terutama untuk data dengan dimensi data target yang cukup besar karena dapat mengurangi nilai *error* secara lebih optimal.

4. KESIMPULAN

Dalam kasus regresi multitarget kedua metode dapat menghasilkan rata-rata *error* yang lebih rendah dari 0.8, hal ini menunjukkan simpangan dari hasil regresi tidak akan lebih dari 1 satuan untuk masing-masing *dataset* untuk kedua metode tersebut. Perubahan dari metode transformasi data ke dimensi yang lebih rendah hingga 88% total dimensi aslinya menggunakan metode transformasi PCA daripada transformasi data ke dimensi yang lebih tinggi dapat menghindarkan model regresi dari *curse of dimensionality*. Selain itu, proses invers dari metode transformasi menggunakan PCA tidak menggunakan metode estimasi *psuedoinverse.* Kedua hal tersebut menunjukkan lebih unggulnya metode transformasi PCA dibanding dengan metode transformasi RLC yang dapat ditunjukkan dari lebih rendahnya rata-rata nilai *error* ketika dibandingkan serta untuk masing-masing *dataset*, metode PCA lebih unggul dalam 7 *dataset* dibandingkan dengan RLC yang hanya lebih unggul dalam 5 *dataset.*

Berdasarkan simpulan yang sudah dijabarkan di atas, metode PCA dapat digunakan dalam meningkatkan performa dari model *regresi* khususnya untuk *dataset* yang memiliki target multidimensi. Saran penelitian lanjutan dari hasil penelitian ini adalah dengan menguji coba kedua metode pada model regresi yang lebih kuat seperti *neural network, deep regression, support vector regression,* dan model regresi lainnya yang mampu melakukan regresi untuk data non-linier dengan lebih baik.

Ucapan Terima Kasih

Penulis mengucapkan terima kasih sebesar-besarnya atas kesempatan yang diberikan oleh LPPM Universitas Katolik Soegijapranata yang telah mendukung secara finansial untuk dapat melakukan penelitian dan menghasilkan tulisan ini.

DAFTAR PUSTAKA

[1] C. Wallisch *et al.*, “Review of guidance papers on regression modeling in statistical series of medical journals,” *PLoS One*, vol. 17, no. 1, p. e0262918, Jan. 2022, doi: 10.1371/JOURNAL.PONE.0262918.

[2] Anila. M and G. Pradeepini, “Least Square Regression for Prediction Problems in Machine Learning using R,” *International Journal of Engineering & Technology*, vol. 7, no. 3.12, pp. 960–962, Jul. 2018, doi: 10.14419/IJET.V7I3.12.17612.

[3] T. Nabarian, M. Aris Ganiardi, and R. F. Malik, “Implementasi Metode Hibrid Fuzzy C-Means dan Fuzzy Swarm untuk Pengelompokkan Data Benang Perusahaan Tekstil,” *Jurnal Teknologi Terpadu*, vol. 6, no. 1, pp. 39–45, Jul. 2020, doi: 10.54914/JTT.V6I1.247.

[4] Carudin, “Pemanfaatan Data Transaksi untuk Dasar membangun Strategi berdasarkan Karakteristik Pelanggan dengan Algoritma K-Means Clustering dan Model RFM,” *Jurnal Teknologi Terpadu*, vol. 7, no. 1, pp. 7–14, Jul. 2021, doi: 10.54914/JTT.V7I1.318.

[5] J. N. Hussain, “High dimensional data challenges in estimating multiple linear regression,” *J Phys Conf Ser*, vol. 1591, no. 1, p. 12035, 2020, doi: 10.1088/1742-6596/1591/1/012035.

[6] I. H. Sarker, “Machine Learning: Algorithms, Real-World Applications and Research Directions,” *SN Comput Sci*, vol. 2, no. 3, pp. 1–21, May 2021, doi: 10.1007/S42979-021-00592-X/FIGURES/11.

[7] S. Jameel and S. Schockaert, “Modeling context words as regions: An ordinal regression approach to word embedding,” *CoNLL 2017 - 21st Conference on Computational Natural Language Learning, Proceedings*, pp. 123–133, 2017, doi: 10.18653/V1/K17-1014.

[8] S. A. T. al Azhima, D. Darmawan, N. F. A. Hakim, I. Kustiawan, M. al Qibtiya, and N. S. Syafei, “Hybrid Machine Learning Model untuk memprediksi Penyakit Jantung dengan Metode Logistic Regression dan Random Forest,” *Jurnal Teknologi Terpadu*, vol. 8, no. 1, pp. 40–46, Jul. 2022, doi: 10.54914/JTT.V8I1.539.

[9] M. Bataineh and T. Marler, “Neural network for regression problems with reduced training sets,” *Neural Networks*, vol. 95, pp. 1–9, 2017, doi: https://doi.org/10.1016/j.neunet.2017.07.018.

[10] S. Lathuilière, P. Mesejo, X. Alameda-Pineda, and R. Horaud, “A Comprehensive Analysis of Deep Regression,” *IEEE Trans Pattern Anal Mach*

Intell, vol. 42, no. 9, pp. 2065–2081, 2020, doi: 10.1109/TPAMI.2019.2910523.

[11] D. Rügamer *et al.*, "deepregression: a Flexible Neural Network Framework for Semi-Structured Deep Distributional Regression," *arXiv:2104.02705 [cs, stat]*, 2021, [Online]. Available: http://arxiv.org/abs/2104.02705

[12] G. Tsoumakas, E. Spyromitros-Xioufis, A. Vrekou, and I. Vlahavas, "Multi-Target Regression via Random Linear Target Combinations," *arXiv:1404.5065 [cs]*, vol. 8726, pp. 225–240, 2014, doi: 10.1007/978-3-662-44845-8_15.

[13] Q. Zhao, E. Adeli, N. Honnorat, T. Leng, and K. M. Pohl, "Variational AutoEncoder for Regression: Application to Brain Aging Analysis," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11765 LNCS, pp. 823–831, 2019, doi: 10.1007/978-3-030-32245-8_91/COVER.

[14] Muthukrishnan R and Maryam Jamila S, "Predictive Modeling Using Support Vector Regression," *International Journal of Scientific & Technology Research*, vol. 9, no. 2, pp. 4863–4865, Feb. 2020, Accessed: Dec. 06, 2022. [Online]. Available: www.ijstr.org

[15] K. Berggren *et al.*, "Roadmap on emerging hardware and technology for machine learning," *Nanotechnology*, vol. 32, no. 1, p. 012002, Oct. 2020, doi: 10.1088/1361-6528/ABA70F.

[16] W. Chiang, X. Liu, T. Zhang, and B. Yang, "A Study of Exact Ridge Regression for Big Data," *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*, pp. 3821–3830, Jan. 2019, doi: 10.1109/BIGDATA.2018.8622274.

[17] D. Xu, Y. Shi, I. W. Tsang, Y.-S. Ong, C. Gong, and X. Shen, "A Survey on Multi-output Learning," Jan. 2019, doi: 10.48550/arxiv.1901.00248.

[18] P. Boye, D. Mireku-Gyimah, and C. A. Okpoti, "Multiple Linear Regression Model for Estimating the Price of a Housing Unit," *Ghana Mining Journal*, vol. 17, no. 2, pp. 66–77, 2017, doi: 10.4314/gm.v17i2.9.

[19] N. Herawati, K. Nisa, E. Setiawan, N. Nusyirwan, and T. Tiryono, "Regularized multiple regression methods to deal with severe multicollinearity," *Int J Stat Appl*, vol. 8, no. 4, pp. 167–172, 2018.

[20] O. Eguasa, E. Edionwe, and J. I. Mbegbu, "Local Linear Regression and the problem of dimensionality: a remedial strategy via a new locally adaptive bandwidths selector," *https://doi.org/10.1080/02664763.2022.2026895*, 2022, doi: 10.1080/02664763.2022.2026895.

[21] Y. Xu, S. Balakrishnan, A. Singh, and A. Dubrawski, "Regression with Comparisons: Escaping the Curse of Dimensionality with Ordinal Information," *Journal of Machine Learning Research*, vol. 21, no. 162, pp. 1–54, 2020, [Online]. Available: http://jmlr.org/papers/v21/19-505.html

[22] T. Górecki and M. Łuczak, "Stacked Regression With a Generalization of the Moore-Penrose Pseudoinverse," *Statistics in Transition New Series*, vol. 18, no. 3, pp. 443–458, 2017, doi: doi:10.21307/stattrans-2016-080.

[23] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimed Tools Appl*, vol. 80, no. 5, pp. 8091–8126, Feb. 2021, doi: 10.1007/S11042-020-10139-6/FIGURES/8.

[24] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[25] C. R. Harris *et al.*, "Array programming with NumPy," *Nature 2020 585:7825*, vol. 585, no. 7825, pp. 357–362, Sep. 2020, doi: 10.1038/s41586-020-2649-2.

[26] A. Asuncion and D. Newman, "UCI: Machine Learning Repository : Solar Flare Dataset," 1989. http://archive.ics.uci.edu/ml/datasets/Solar+Flare (accessed Jun. 05, 2022).

[27] S. Džeroski, D. Demsar, and J. Grbović, "Predicting Chemical Parameters of River Water Quality from Bioindicator Data," *Applied Intelligence*, vol. 13, pp. 7–17, 2000, doi: 10.1023/A:1008323212047.

[28] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves, and I. Vlahavas, "Multi-target regression via input space expansion: treating targets as inputs," *Mach Learn*, vol. 104, no. 1, pp. 55–98, 2016, doi: 10.1007/s10994-016-5546-z.

[29] A. Karalic and I. Bratko, "First Order Regression," *Mach Learn*, vol. 26, pp. 147–176, 1997, doi: 10.1023/A:1007365207130.