JSI (Jurnal Sistem Informasi) Universitas Dirgantara Marsekal Suryadarma
p-ISSN: 2355-9675
e-ISSN: 2541-3228

Prediksi Penjualan Produk Sepatu dengan Menggunakan Algoritma K-Nearest Neighbor Regression dan Cross Validation

Ratna Salkiawati[1*], Hendarman Lubis[2], Nurfiyah[3], Fitu Sapril Telaumbanua[4]

[1,2,3,4]Fakultas Ilmu Komputer, Universitas Bhayangkara Jakarta Raya, Indonesia
ratna_tind@dsn.ubharajaya.ac.id; hendarman.lubis@dsn.ubharajaya.ac.id; nurfiyah@dsn.ubharajaya.ac.id; fitu.sapril17@mhs.ubharajaya.ac.id

Article Info

Article history:

Received November 15, 2024
Accepted Desember 16, 2024
Published January 2, 2025

Kata Kunci:

K-Nearest Neighbor,
Regression,
Prediksi,
Knowledge discovery in database,
RMSE

ABSTRAK

Penjualan sepatu di Toko "X" mencakup berbagai merek, seperti Fladeo, Cardinal, dr. Kevin, dan Jackson. Sistem pengelolaan data penjualan di toko ini pada saat penelitian masih menggunakan pencatatan secara manual, di mana hasil penjualan hanya diproses dalam format MS Excel. Untuk memudahkan pengelolaan dan perencanaan penjualan di masa depan, diperlukan prediksi penjualan menggunakan teknik klasifikasi data mining, yaitu algoritma K-Nearest Neighbor Regression. Berdasarkan hasil penelitian, prediksi penjualan sepatu terlaris menunjukkan bahwa nilai K = 2 menghasilkan RMSE 0,43 untuk produk Fladeo, K = 3 menghasilkan RMSE 0,46 untuk produk Cardinal, K = 13 menghasilkan RMSE 0,46 untuk produk dr. Kevin, dan K = 6 menghasilkan RMSE 0,49 untuk produk Jackson. Berdasarkan pedoman RMSE, dapat disimpulkan bahwa semua model yang diuji menunjukkan tingkat kesalahan sedang, yaitu antara 0,30 hingga 0,56.

Corresponding Author:

Ratna Salkiawati,
Faculty of Computer Science,
Universitas Bhayangkara Jakarta Raya
Email: *ratna_tind@dsn.ubharajaya.ac.id

1. PENDAHULUAN

1.1 Latar Belakang

Toko "X" adalah tempat perbelanjaan yang menawarkan berbagai merek sepatu ternama seperti Fladeo, Cardinal, dr. Kevin, dan Jackson, dengan berbagai model sepatu yang stylish dan modern. Toko "X" terus berupaya untuk meningkatkan dan mengembangkan bisnisnya agar dapat memenuhi kebutuhan pelanggan dalam hal kualitas sepatu, harga, dan pemasaran. Mengingat persaingan yang semakin ketat, penting bagi toko ini untuk memastikan produk sepatu mereka dikenal oleh konsumen. Perbedaan dalam atribut sepatu memungkinkan konsumen untuk mengetahui kelebihan dan kekurangan masing-masing produk. Kepuasan pelanggan terhadap sepatu yang dibeli menjadi faktor kunci dalam merebut pangsa pasar (Awaludin & Mantik, 2023). Namun, saat ini system pengelolaan data penjualan di toko ini masih menerapkan metode manual, dengan mencatat data dilakukan menggunakan Microsoft Excel. Hal ini menyebabkan penumpukan data yang tidak terorganisir dengan baik, sehingga mempersulit pemahaman dan pengembangan informasi terkait penjualan sepatu setiap tahunnya. Oleh karena itu, diperlukan prediksi penjualan sepatu untuk tahun yang akan datang.

Dalam penelitian sebelumnya yang berjudul Perbandingan Algoritma K-Nearest Neighbor dengan

Decision Tree Dalam Memprediksi Penjualan Makanan Hewan Peliharaan Di Petshop Dore Vet Clinic, dengan menggunakan metode K-NN, dari 30 data yang dianalisis, terdapat beberapa kondisi yang dihasilkan. Sebanyak 6 data diklasifikasikan sebagai produk terlaris sesuai dengan prediksi K-NN, namun 3 dari 6 produk yang diprediksi terlaris ternyata tidak terlaris (dengan urutan data 1, 2, 6). Sedangkan 24 data lainnya diprediksi tidak terlaris, namun 10 data dari prediksi tersebut ternyata terlaris (urutan data 22, 5, 16, 26, 28, 19, 17, 20, 23, 24). Menggunakan metode decision tree dengan algoritma C4.5, diketahui bahwa dari 30 data, merek Purina termasuk produk terlaris, sementara ada 4 data dari Royal Canin yang termasuk dalam kategori false negative (Meliala & Hasugian, 2020).

Pada penelitian lainnya yang berjudul Prediksi Harga Beras Premium dengan Metode Algoritma K-Nearest Neighbor, hasil penelitian ini menunjukkan bahwa metode K-Nearest Neighbor dengan model regresi dapat memprediksi harga beras untuk tahun 2014 hingga 2019 dengan nilai RMSE sebesar 0,125 dan parameter K = 2 setelah dilakukan normalisasi (Mukhlisin et al., 2020).

Penelitian serupa lainnya juga dilakukan oleh Cholil et al. (2021), Dewi (2016), Harahap & Sulindawaty (2020), Hutami & Astuti (2016), Puspita Hidayanti (2020), Rahman et al. (2018), Reza Noviansyah et al. (2018), Sasmita Susanto & Al Fatta (2018), dan Yustanti (2012).

1.2 Teori

1. *K-Nearest Neigbor*

Metode K-NN merupakan suatu metode untuk melakukan klasifikasi terhadap suatu obyek yang berdasar kepada data training yang mempunyai jarak yang paling dekat dari obyek tersebut. Algoritma K-NN Regresi adalah algoritma dengan melakukan pengelompokkan data dengan berdasarkan letak ketetanggaannya, pengelompokan data tersebut bergantung pada jumlah nilai K dan nilai RMSE dihasilkan untuk melakukan pengukuran tingkat *error* dari model yang sudah dibuat (Kotu & Deshpande, n.d.; Ye, 2014).

Pengertian lain tentang K-NN adalah sebuah metode untuk melakukan klasifikasi terhadap objek berdasarkan dengan data pembelajaran (*neighbor*) yang jaraknya paling dekat dengan objek tersebut (Russell, 2018). Dekat atau jauhnya *neighbor* biasanya dihitung dengan berdasarkan jarak *Euclidean* sehingga diperlukan suatu sistem klasifikasi sebagai sebuah sistem yang mampu mencari informasi klasifikasi suatu objek. Metode K-NN dibagi menjadi dua fase, yaitu pembelajaran (*training*) dan klasifikasi atau pengujian (*testing*). Pada fase pembelajaran, algoritma ini biasanya hanya melakukan penyimpanan pada vektor-vektor fitur dan melakukan klasifikasi dari data pembelajaran yang telah dibuat. Pada fase klasifikasi, fitur-fitur yang sama dilakukan perhitungan untuk data yang akan dilakukan uji coba (yang klasifikasinya tidak diketahui). Selanjutnya jarak dari vektor yang baru ini terhadap seluruh vektor data pembelajaran dihitung, dan sejumlah nilai k buah *neighbor* yang paling dekat diambil (Baharuddin et al., 2019).

Adapun perhitungan *euclidean distance* menggunakan persamaan sebagai berikut:

$$d_i = \sqrt{\sum_{i=1}^{n} (x_{2i} - x_{1i})^2}$$

Keterangan :

D	: jarak terdekat
x1	: sampel data atau data training
x2	: data uji atau data testing
I	: atribut data dari 1 sampai n
N	: jumlah atribut setiap kasus

2. *K-Fold Cross Validation*

K-fold cross validation mirip dengan metode *subsampling* yang melakukan acak berulang, tetapi pengambilan sampel dilakukan sedemikian rupa sehingga tidak ada dua *set tes* yang tumpang tindih. Dalam *k-fold cross validation*, *learning set* yang tersedia dipartisi menjadi k subset yang terpisah dengan ukuran yang kira-kira sama. Kata “lipatan” mengacu pada jumlah himpunan bagian yang dihasilkan. Partisi ini dilakukan dengan mengambil sampel kasus secara acak dari set pembelajaran tanpa pengembalian. Model dilatih menggunakan k 1 sebagai himpunan bagian, yang bersama-sama mewakili

himpunan pelatihan. Kemudian, model diterapkan ke *subset* yang tersisa, dengan dilambangkan sebagai *set validasi*, dan pengukuran kinerjanya. Prosedur ini diulang sampai masing-masing k subset telah berfungsi sebagai *set validasi* (Berrar, 2018).

Kesalahan generalisasi pada *K-fold cross validation*, dan ditetapkan K menjadi 10 dengan dua alasan yaitu (Nikmatun & Waspada, 2019):

1. untuk menyeimbangkan antara biaya komputasi dan estimasi yang diandalkan
2. untuk perbandingan yang adil pada data latih dan data uji

Untuk 10-*fold cross validation*, dataset dibagi menjadi 10 lipatan yang saling terpisah dengan ukuran yang hampir sama. Dalam setiap *run*, 9 subset digunakan untuk pelatihan dan sisanya untuk validasi .

3. Root Mean Square Error (RMSE)

RMSE merupakan proses pengecekan kesalahan yang melakukan perbandingan nilai sesungguhnya dengan nilai yang didapatkan dari pengujian dengan hasil dinyatakan sebagai nilai mutlak yang didapat berkisar dari 0 hingga ∞ (Mailund, 2017). Nilai RMSE yang dapat dihitung dengan persamaan sebagai berikut:

$$RMSE = \sqrt{\sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2}$$

Keterangan :
RMSE : nilai root mean square error
Y : nilai hasil observasi
Ŷ : nilai hasil prediksi
I : urutan data
N : jumlah data

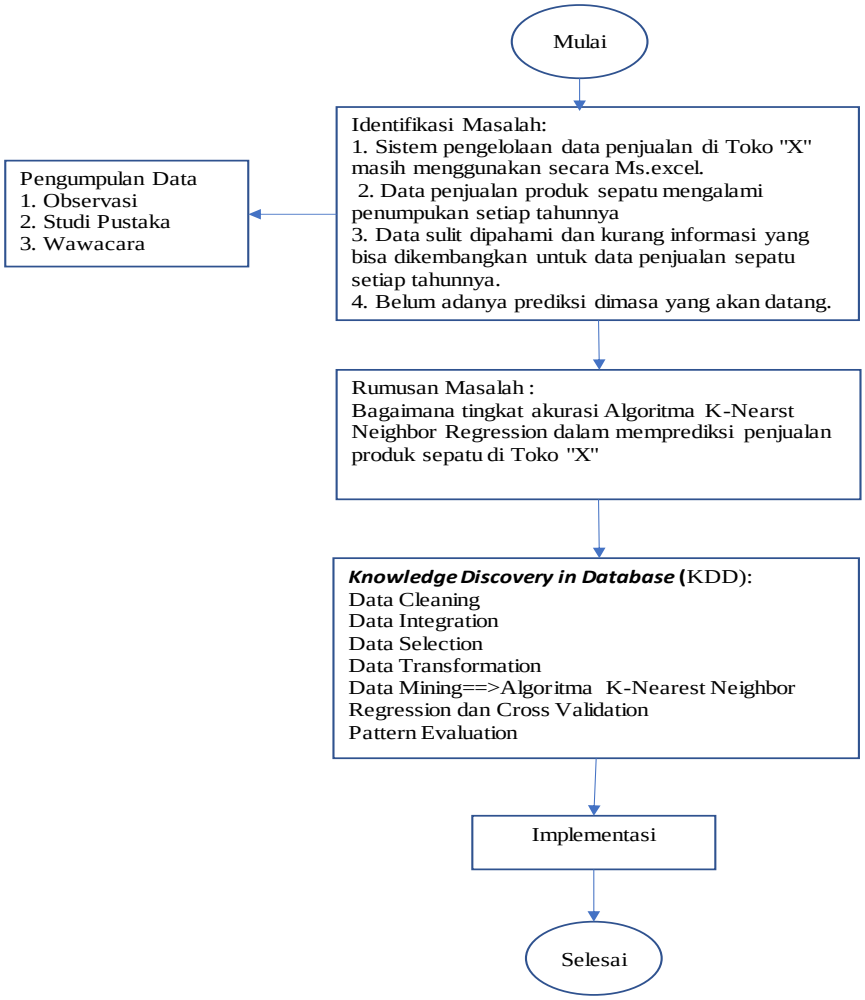
Nilai RMSE merupakan nilai rata-rata dari besar kesalahan pada suatu sampel data. Semakin besar nilai RMSE maka semakin besar pula tingkat perbedaan kesalahan pada masing-masing sample data yang dihitung. Untuk memudahkan dalam memahami nilai RMSE maka digunakan tabel dibawah ini:

Tabel 1. Pedoman Interpretasi RMSE

RMSE	**Tingkat Kesalahan**
0,00 – 0,299	Kecil
0,30 – 0,599	Sedang
0,60 – 0,899	Besar
>0,90	Sangat Besar

4. METODE

Berikut ini kerangka penelitian:



Gambar 1. Kerangka Penelitian di Toko ‘X”

Berikut adalah penjelasan kerangka penelitian:

a. Mulai: melakukan persiapan sebelum melakukan penelitian.
b. Identifikasi Masalah: peneliti melakukan pengumpulan data dan berhasil mendapatkan identifikasi masalah Toko “X”
c. Rumusan Masalah: menentukan identifikasi masalah akan mendapatkan apa yang dibuat dalam perumusan masalah.
d. Metode Pengumpulan Data: tahapan dalam metode pengumpulan data dalam penelitian ini yaitu observasi, studi pustaka, dan wawancara.
e. Metode Perancangan: metode yang ditentukan dalam penelitian ini yaitu *Knowledge Discovery in Database* dengan menggunakan teknik *K-Nearest Neighbor Regression*.
f. Penerapan metode K-Nearest Neighbor Regession untuk mempredikasi prodk sepatu terlaris .
g. Pengujian : model di uji dengan menggunakan teknik RMSE
h. Selesai : didapat hasil prediksi

5. HASIL DAN PEMBAHASAN

3.1 Pembentukan Dataset

Setelah pengolahan data dilakukan dan menghasilkan input penjualan produk dari selama 3 tahun, kemudian data tersebut akan digunakan untuk dilakukan training. Data training dikelompokan menjadi 2 bagian yaitu data input dan data target. Data input merupakan data penjualan dari bulan ke-1 sampai bulan ke-12, sedangkan data target menggunakan data bulan ke-13.

Tabel 2. Data Training

Data Input												Target
96	69	43	84	57	57	40	31	64	98	67	77	66
69	43	84	57	57	40	31	64	98	67	77	66	45
43	84	57	57	40	31	64	98	67	77	66	45	60
84	57	57	40	31	64	98	67	77	66	45	60	80
57	57	40	31	64	98	67	77	66	45	60	80	55
57	40	31	64	98	67	77	66	45	60	80	55	56
40	31	64	98	67	77	66	45	60	80	55	56	61
31	64	98	67	77	66	45	60	80	55	56	61	63
64	98	67	77	66	45	60	80	55	56	61	63	54
98	67	77	66	45	60	80	55	56	61	63	54	59
67	77	66	45	60	80	55	56	61	63	54	59	52
77	66	45	60	80	55	56	61	63	54	59	52	68
66	45	60	80	55	56	61	63	54	59	52	68	108
45	60	80	55	56	61	63	54	59	52	68	108	117
60	80	55	56	61	63	54	59	52	68	108	117	93
80	55	56	61	63	54	59	52	68	108	117	93	91
55	56	61	63	54	59	52	68	108	117	93	91	117
56	61	63	54	59	52	68	108	117	93	91	117	74
61	63	54	59	52	68	108	117	93	91	117	74	29
63	54	59	52	68	108	117	93	91	117	74	29	27
54	59	52	68	108	117	93	91	117	74	29	27	34
59	52	68	108	117	93	91	117	74	29	27	34	30
52	68	108	117	93	91	117	74	29	27	34	30	34
68	108	117	93	91	117	74	29	27	34	30	34	31

Sumber: Hasil Penelitian

3.2 Normalisasi Dataset

Kemudian proses normalisasi data, dilakukan dengan cara membuat data yang sudah ada menjadi nilai yang lebih kecil. Data hasil penelitian yang sudah diolah dinormalisasi dengan menjadikan data menjadi jarak [0, 1], yang artinya nilai minimal dari data tersebut menjadi 0 dan nilai maksimal dari data menjadi 1 sehingga data diantara minimal dan maksimal menyesuaikan antara jarak yang digunakan.

```
from sklearn.preprocessing import MinMaxScaler
# default range 0-1
mmscaler = MinMaxScaler()
dataset_norm = mmscaler.fit_transform(df)
pd.DataFrame(dataset_norm)
```

Gambar 1. Script Normalisasi Training

Hasil Implementasi Normalisasi dataset:

0	0.970149	0.493506	0.139535	0.616279	0.302326	0.302326	0.104651	0.022727	0.411111	0.788889	0.444444	0.555556	0.433333
1	0.567164	0.155844	0.616279	0.302326	0.302326	0.104651	0.000000	0.397727	0.788889	0.444444	0.555556	0.433333	0.200000
2	0.179104	0.688312	0.302326	0.302326	0.104651	0.000000	0.383721	0.784091	0.444444	0.555556	0.433333	0.200000	0.366667
3	0.791045	0.337662	0.302326	0.104651	0.000000	0.383721	0.779070	0.431818	0.555556	0.433333	0.200000	0.366667	0.588889
4	0.388060	0.337662	0.104651	0.000000	0.383721	0.779070	0.418605	0.545455	0.433333	0.200000	0.366667	0.588889	0.311111
5	0.388060	0.116883	0.000000	0.383721	0.779070	0.418605	0.534884	0.420455	0.200000	0.366667	0.588889	0.311111	0.322222
6	0.134328	0.000000	0.383721	0.779070	0.418605	0.534884	0.406977	0.181818	0.366667	0.588889	0.311111	0.322222	0.377778
7	0.000000	0.428571	0.779070	0.418605	0.534884	0.406977	0.162791	0.352273	0.588889	0.311111	0.322222	0.377778	0.400000
8	0.492537	0.870130	0.418605	0.534884	0.406977	0.162791	0.337209	0.579545	0.311111	0.322222	0.377778	0.400000	0.300000
9	1.000000	0.467532	0.534884	0.406977	0.162791	0.337209	0.569767	0.295455	0.322222	0.377778	0.400000	0.300000	0.355556
10	0.537313	0.597403	0.406977	0.162791	0.337209	0.569767	0.279070	0.306818	0.377778	0.400000	0.300000	0.355556	0.277778
11	0.686567	0.454545	0.162791	0.337209	0.569767	0.279070	0.290698	0.363636	0.400000	0.300000	0.355556	0.277778	0.455556
12	0.522388	0.181818	0.337209	0.569767	0.279070	0.290698	0.348837	0.386364	0.300000	0.355556	0.277778	0.455556	0.900000
13	0.208955	0.376623	0.569767	0.279070	0.290698	0.348837	0.372093	0.284091	0.355556	0.277778	0.455556	0.900000	1.000000
14	0.432836	0.636364	0.279070	0.290698	0.348837	0.372093	0.267442	0.340909	0.277778	0.455556	0.900000	1.000000	0.733333
15	0.731343	0.311688	0.290698	0.348837	0.372093	0.267442	0.325581	0.261364	0.455556	0.900000	1.000000	0.733333	0.711111
16	0.358209	0.324675	0.348837	0.372093	0.267442	0.325581	0.244186	0.443182	0.900000	1.000000	0.733333	0.711111	1.000000
17	0.373134	0.389610	0.372093	0.267442	0.325581	0.244186	0.430233	0.897727	1.000000	0.733333	0.711111	1.000000	0.522222
18	0.447761	0.415584	0.267442	0.325581	0.244186	0.430233	0.895349	1.000000	0.733333	0.711111	1.000000	0.522222	0.022222
19	0.477612	0.298701	0.325581	0.244186	0.430233	0.895349	1.000000	0.727273	0.711111	1.000000	0.522222	0.022222	0.000000
20	0.343284	0.363636	0.244186	0.430233	0.895349	1.000000	0.720930	0.704545	1.000000	0.522222	0.022222	0.000000	0.077778
21	0.417910	0.272727	0.430233	0.895349	1.000000	0.720930	0.697674	1.000000	0.522222	0.022222	0.000000	0.077778	0.033333
22	0.313433	0.480519	0.895349	1.000000	0.720930	0.697674	1.000000	0.511364	0.022222	0.000000	0.077778	0.033333	0.077778
23	0.552239	1.000000	1.000000	0.720930	0.697674	1.000000	0.500000	0.000000	0.000000	0.077778	0.033333	0.077778	0.044444

Gambar 2. Sampel Hasil Normalisasi

Setelah data dinormalisasi maka dilakukan pembagian data kembali. Pembentukan variabel untuk data input adalah huruf X dengan data dimulai dari index ke-0 sampai ke-11, sedangkan data target menggunakan huruf Y dengan data yang merupakan index ke-12.

```
X = dataset_norm[:,[0,1,2,3,4,5,6,7,8,9,10,11]]
y_norm = dataset_norm[:, [-1]]
y = y_norm.reshape(24)
```

Gambar 3. Script Pembagian Data Input dan Target Training

	0	1	2	3	4	5	6	7	8	9	10	11
0	0.000000	0.123077	0.430769	0.046154	0.061538	0.138462	0.386364	0.300000	0.355556	0.277778	0.455556	0.900000
1	0.238095	0.430769	0.046154	0.061538	0.138462	0.169231	0.284091	0.355556	0.277778	0.455556	0.900000	1.000000
2	0.555556	0.046154	0.061538	0.138462	0.169231	0.030769	0.340909	0.277778	0.455556	0.900000	1.000000	0.733333
3	0.158730	0.061538	0.138462	0.169231	0.030769	0.107692	0.261364	0.455556	0.900000	1.000000	0.733333	0.711111
4	0.174603	0.138462	0.169231	0.030769	0.107692	0.000000	0.443182	0.900000	1.000000	0.733333	0.711111	1.000000
5	0.253968	0.169231	0.030769	0.107692	0.000000	0.246154	0.897727	1.000000	0.733333	0.711111	1.000000	0.522222
6	0.285714	0.030769	0.107692	0.000000	0.246154	0.861538	1.000000	0.733333	0.711111	1.000000	0.522222	0.022222
7	0.142857	0.107692	0.000000	0.246154	0.861538	1.000000	0.727273	0.711111	1.000000	0.522222	0.022222	0.000000
8	0.222222	0.000000	0.246154	0.861538	1.000000	0.630769	0.704545	1.000000	0.522222	0.022222	0.000000	0.077778
9	0.111111	0.246154	0.861538	1.000000	0.630769	0.600000	1.000000	0.522222	0.022222	0.000000	0.077778	0.033333
10	0.365079	0.861538	1.000000	0.630769	0.600000	1.000000	0.511364	0.022222	0.000000	0.077778	0.033333	0.077778
11	1.000000	1.000000	0.630769	0.600000	1.000000	0.338462	0.000000	0.000000	0.077778	0.033333	0.077778	0.044444

Gambar 4. Sampel Hasil Normalisasi Penjualan

3.3 *K-Fold Cross Validation*

Dalam tahap ini digunakan *K-Fold Cross Validation* untuk membagi serta melakukan validasi

data. Kemudian hasil *split* yang dilakukan oleh *K-Fold Cross Validation* sebanyak n. Penelitian ini menggunakan nilai n = 10. Implementasi pada program sebagai berikut:

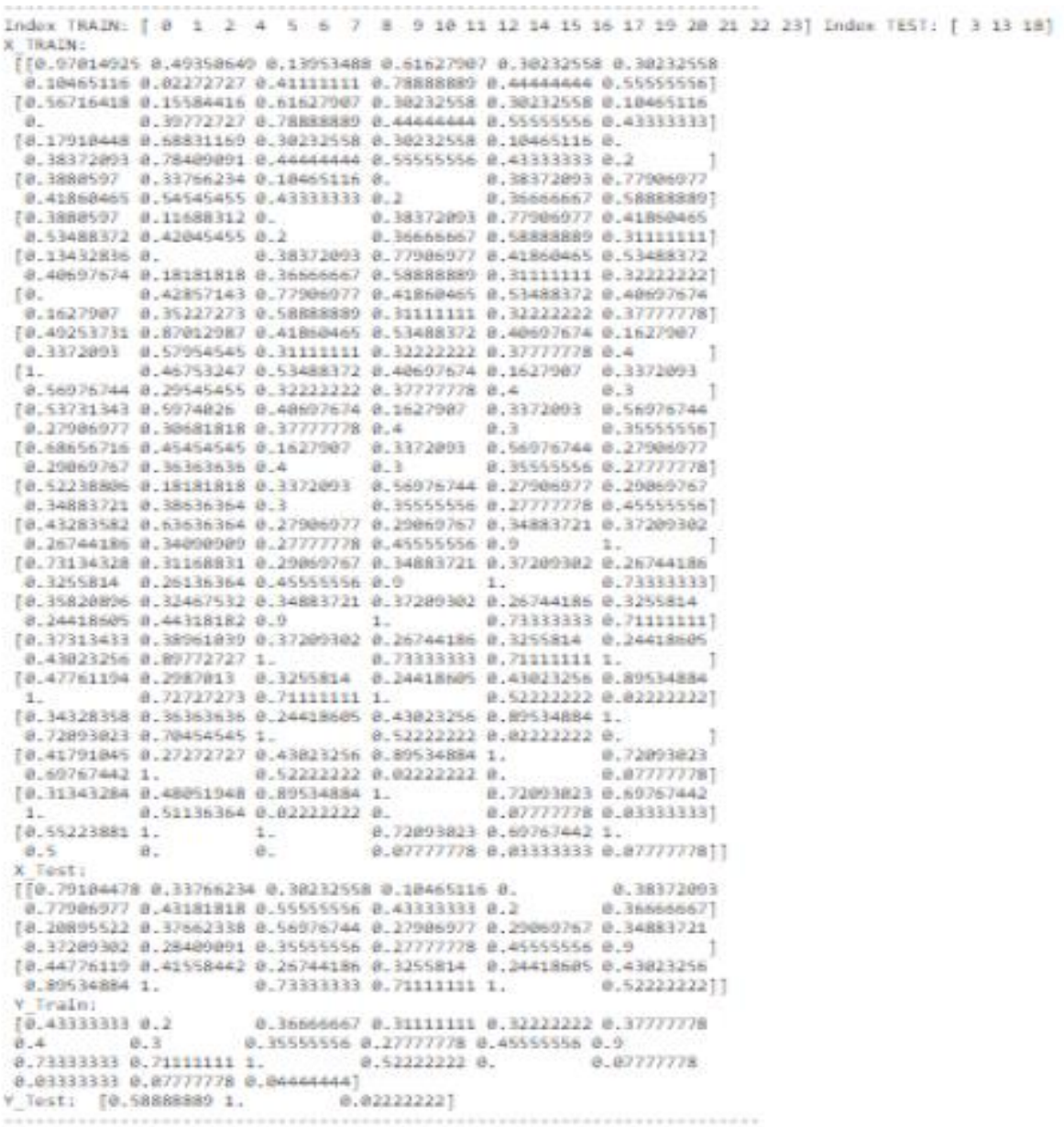
```
TRAIN: [ 0  1  2  4  5  6  7  8  9 10 11 12 14 15 16 17 19 20 21 22 23] TEST: [ 3 13 18]
TRAIN: [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 15 16 18 19 21 22 23] TEST: [14 17 20]
TRAIN: [ 0  1  3  5  6  7  8  9 11 12 13 14 15 16 17 18 19 20 21 22 23] TEST: [ 2  4 10]
TRAIN: [ 0  1  2  3  4  5  8  9 10 11 12 13 14 15 16 17 18 20 21 22 23] TEST: [ 6  7 19]
TRAIN: [ 0  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 22 23] TEST: [ 1 21]
TRAIN: [ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 17 18 19 20 21 22 23] TEST: [ 0 16]
TRAIN: [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 16 17 18 19 20 21 22] TEST: [15 23]
TRAIN: [ 0  1  2  3  4  5  6  7  8 10 11 12 13 14 15 16 17 18 19 20 21 23] TEST: [ 9 22]
TRAIN: [ 0  1  2  3  4  5  6  7  9 10 11 13 14 15 16 17 18 19 20 21 22 23] TEST: [ 8 12]
TRAIN: [ 0  1  2  3  4  6  7  8  9 10 12 13 14 15 16 17 18 19 20 21 22 23] TEST: [ 5 11]
```

Gambar 5. Hasil Implementasi *K-Fold Cross Validation*

Split yang tampil masih berupa dataset dalam bentuk *index* sehingga untuk melihat data training yang telah di-*split* menggunakan *script* sebagai berikut:

```
for train_index, test_index in cv.split(X):
   print("-------------------------------------------------------------------------")
   print("Index TRAIN:", train_index, "Index TEST:", test_index)
   X_train, X_test = X[train_index], X[test_index]
   y_train, y_test = y[train_index], y[test_index]
   print("X_TRAIN: \n",X_train,"\n X_Test: \n",X_test,"\n Y_Train: \n", y_train,"\nY_Test: ", y_test)
   print("-------------------------------------------------------------------------\n\n")
```

Gambar 6. Data Training dan Testing



Gambar 7. Hasil Split K-Fold Cross Validation

Hasil diatas merupakan hasil *split* pada iterasi pertama yang menampilkan data *training* x, *test* x, serta data *training* y dan *data test* y.

3.4 *K-Nearest Neighbor Regression*

Model *K-Nearest Neighbor Regression* dibangun sebagai Sampel pengujian model yang akan digunakan pada evaluasi dengan menggunakan *Cross Validation*.

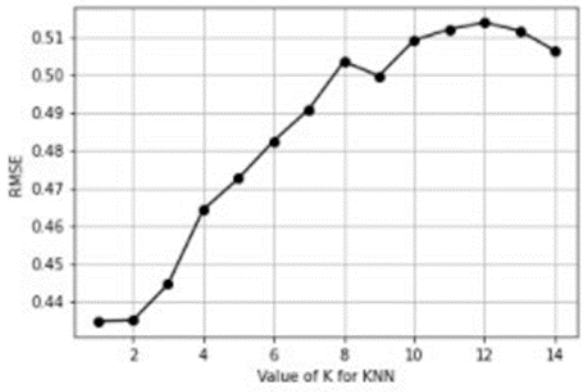
```
## membangun model
model = KNeighborsRegressor(n_neighbors=3, metric='euclidean')
model
```

```
KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='euclidean',
                    metric_params=None, n_jobs=None, n_neighbors=3, p=2,
                    weights='uniform')
```

Gambar 8. Implementasi K-NN dan Cross Validation

3.5 Evaluasi

Evaluasi ini dilakukan dengan melakukan pengujian model menggunakan *cross validation* untuk setiap nilai k dari *k-nearest neighbor* dalam *range* 1-15.



Gambar 9. Grafik Nilai RMSE

Grafik tersebut menunjukkan bahwa ada perbedaan nilai RMSE berdasarkan nilai k pada *k-nearest neighbor*. Nilai k tersebut sangat mempengaruhi hasil dari keakuratan model prediksi yang dilakukan.

3.6 Prediksi

Dari nilai RMSE didapatkan nilai akurasi terbaik untuk setiap produk.

Tabel 2. Hasil nilai akurasi terbaik

Produk	Nilai K terbaik	RMSE terbaik	Prediksi Penjualan
Fladeo	3	0,43494	Maret
Cardinal dr.	3	0,46214	Januari
Kevin	13	0,45539	Juli
Jackson	14	0,48781	November

6. KESIMPULAN

Berdasarkan hasil penelitian dan pengujian model prediksi penjualan menggunakan metode K-Nearest Neighbor Regression, dapat disimpulkan bahwa nilai k yang paling optimal dari rentang 1 hingga 15 adalah sebagai berikut: k = 2 untuk produk sepatu Fladeo dengan RMSE sebesar 0.43494, k = 3 untuk produk Cardinal dengan RMSE sebesar 0.46214, k = 13 untuk produk dr. Kevin dengan RMSE sebesar 0.45539, dan k = 14 untuk produk Jackson dengan RMSE sebesar 0.48789. Mengacu pada pedoman interpretasi RMSE, dapat disimpulkan bahwa tingkat kesalahan seluruh model yang diuji memiliki tingkat kesalahan sedang, karena nilai RMSE yang diperoleh berada dalam rentang 0,30 hingga 0,599.

DAFTAR PUSTAKA

Awaludin, M., & Mantik, H. (2023). Penerapan Metode Servqual Pada Skala Likert Untuk Mendapatkan Kualitas Pelayanan Kepuasan Pelanggan. *Jurnal Sistem Informasi Univesitas Suryadarma*, *10*(1).

Baharuddin, M. M., Azis, H., & Hasanuddin, T. (2019). ANALISIS PERFORMA METODE K-NEAREST NEIGHBOR UNTUK IDENTIFIKASI JENIS KACA. *ILKOM Jurnal Ilmiah*, *11*(3), 269–274. https://doi.org/10.33096/ilkom.v11i3.489.269-274

Berrar, D. (2018). Cross-validation. In *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics* (Vols. 1–3, pp. 542–545). Elsevier. https://doi.org/10.1016/B978-0-12-809633-8.20349-X

Cholil, S. R., Handayani, T., Prathivi, R., & Ardianita, T. (2021). Implementasi Algoritma Klasifikasi K-Nearest Neighbor (KNN) Untuk Klasifikasi Seleksi Penerima Beasiswa. In *IJCIT (Indonesian Journal on Computer and Information Technology)* (Vol. 6, Issue 2).

Dewi, S. (2016). KOMPARASI 5 METODE ALGORITMA KLASIFIKASI DATA MINING PADA PREDIKSI KEBERHASILAN PEMASARAN PRODUK LAYANAN PERBANKAN. *Jurnal Techno Nusa Mandiri* .

Harahap, P. N., & Sulindawaty, S. (2020). Implementasi Data Mining Dalam Memprediksi Transaksi Penjualan Menggunakan Algoritma Apriori (Studi Kasus PT.Arma Anugerah Abadi Cabang Sei Rampah). *MATICS*, *11*(2), 46. https://doi.org/10.18860/mat.v11i2.7821

Hutami, R., & Astuti, E. Z. (2016). *Implementasi MetodemK-Nearest Neighbor Untuk Prediksi Penjualan Furniture Pasa CV.Octo Agung Jepara*.

Kotu, V., & Deshpande, B. (n.d.). *Predictive analytics and data mining : concepts and practice with RapidMiner*.

Mailund, T. (2017). Beginning Data Science in R: Data Analysis, Visualization, and Modelling for the Data Scientist. In *Beginning Data Science in R: Data Analysis, Visualization, and Modelling for the Data Scientist*. Apress Media LLC. https://doi.org/10.1007/978-1-4842-2671-1

Meliala, D. M., & Hasugian, P. (2020). Perbandingan Algoritma K-Nearest Neighbor Dengan Decision Tree Dalam Memprediksi Penjualan Makanan Hewan Peliharaan Di Petshop Dore Vet Clinic. *Jurnal Teknologi Informasi*, *3*.

Mukhlisin, Y., Imrona, M., & Murdiansyah, D. T. (2020). *Prediksi Harga Beras Premium dengan Metode Algoritma K-Nearest Neighbor*.

Nikmatun, I. A., & Waspada, I. (2019). IMPLEMENTASI DATA MINING UNTUK KLASIFIKASI MASA STUDI MAHASISWA MENGGUNAKAN ALGORITMA K-NEAREST NEIGHBOR. *Jurnal SIMETRIS*, *10*(2).

Puspita Hidayanti, W. (2020). Penerapan Algoritma K-Nearest Neighbor Untuk Klasifikasi Efektivitas Penjualan Vape (Rokok Elektrik) pada "Lombok Vape On." *Jurnal Informatika Dan Teknologi*, *3*(2).

Rahman, M. A., Hidayat, N., & Supianto, A. A. (2018). *Komparasi Metode Data Mining K-Nearest Neighbor Dengan Naïve Bayes Untuk Klasifikasi Kualitas Air Bersih (Studi Kasus PDAM Tirta Kencana Kabupaten Jombang)* (Vol. 2, Issue 12). http://j-ptiik.ub.ac.id

Reza Noviansyah, M., Rismawan, T., Marisa Midyanti, D., Sistem Komputer, J., & MIPA Universitas Tanjungpura Jl Hadari Nawawi, F. H. (2018). PENERAPAN DATA MINING MENGGUNAKAN METODE K-NEAREST NEIGHBOR UNTUK KLASIFIKASI INDEKS CUACA KEBAKARAN BERDASARKAN DATA AWS (AUTOMATIC WEATHER STATION) (STUDI KASUS: KABUPATEN KUBU RAYA). In *Jurnal Coding, Sistem Komputer Untan* (Vol. 06, Issue 2).

Russell, R. (2018). *Machine Learning: Step-by-Step Guide To Implement Machine Learning Algorithms with Python*.

Sasmita Susanto, E., & Al Fatta, H. (2018). Prediksi KELULUSAN MAHASISWA MAGISTER TEKNIK INFORMATIKA UNIVERSITAS AMIKOM YOGYAKARTA MENGGUNAKAN METODE K-NEAREST NEIGHBOR. *Jurnal Teknologi Informasi*.

Ye, Nong. (2014). *Data mining : theories, algorithms, and examples*. CRC Press.

Yustanti, W. (2012). *Algoritma K-Nearest Neighbour untuk Memprediksi Harga Jual Tanah* (Vol. 9, Issue 1).