BAB V
HASIL DAN PEMBAHASAN

Hasil dan pembahasan dari penelitian terdiri dari pembahasan dari hasil pengujian yang sudah dilakukan oleh peneliti. Oleh karena itu, penjelasan dari hasil penelitian adalah berikut.

5.1 Hasil dan Pembahasan Perencanaan

Di dalam perencanaan, peneliti sudah membuat diagram alur penelitian serta alur pengujian algoritma yang ingin dilakukan. Setelah melakukan penelitian, peneliti mendapatkan hasil yang akan dibahas sebagai berikut ini.

5.1.1 Data *Pre-processing*

Data memasuki tahap *pre-processing* sebelum dilakukan pengujian algoritma terhadap model. Data akan dilakukan normalisasi pada tahap ini. Data sebelum normalisasi ditunjukan pada tabel 5.1

Tabel 5.1 Data Asli

No	Harga	Model	Umur	Jarak Satu	Jarak Dua	Transmisi	Kapasitas Awal	Kapasitas Akhir
1	112000000	102	6	90000	95000.0	1	1000	1500
2	175000000	102	2	15000	20000.0	2	1000	1500
3	112000000	103	5	50000	55000.0	2	1000	1500
4	143000000	104	1	30000	35000.0	1	1000	1500
5	137000000	104	3	30000	35000.0	1	1000	1500
6	123000000	104	5	65000	70000.0	2	1000	1500
7	98000000	103	7	65000	70000.0	2	1000	1500
8	86500000	103	8	190000	195000.0	2	1000	1500

Data asli yang terdapat pada dataset dilengkapi dengan informasi mengenai harga, model, umur, jarak, jenis transmisi, dan kapasitas mesin dari mobil. Tahun produksi dari data mobil yang didapatkan mulai 2004 – 2019. Kemudian data jenis model mobil dan jenis transmisi mobil yang sudah diubah dari non-numerik menjadi numerik. Perubahan data tersebut sudah dijelaskan pada Tabel. 4.2

Peneliti menggunakan *function min-max scaler* di Python dalam melakukan normalisasi data pada tahap *pre-processing*. Hal tersebut bertujuan untuk melakukan penyesuaian data terhadap rentang tertentu (0 dan 1). Hasil dari normalisasi data ditunjukan pada tabel 5.2.

Tabel 5.2 Normalisasi Data

No	Harga	Model	Umur	Jarak Satu	Jarak Dua	Transmisi	Kapasitas Awal	Kapasitas Akhir
1	0.339181	0.333333	0.400000	0.315789	0.315789	0.0	0.0	0.0
2	0.707602	0.333333	0.133333	0.052632	0.052632	1.0	0.0	0.0
3	0.339181	0.666667	0.333333	0.175439	0.175439	1.0	0.0	0.0
4	0.520468	1.000000	0.066667	0.105263	0.105263	0.0	0.0	0.0
5	0.485380	1.000000	0.200000	0.105263	0.105263	0.0	0.0	0.0
6	0.403509	1.000000	0.333333	0.228070	0.228070	1.0	0.0	0.0
7	0.257310	0.666667	0.466667	0.228070	0.228070	1.0	0.0	0.0
8	0.190058	0.666667	0.533333	0.666667	0.666667	1.0	0.0	0.0

5.1.2 Pembagian Data

Data yang sudah melalui tahap *pre-processing* ada sebanyak 781 data. Setelah itu data dibagi menjadi data *training* dan data *test*. Peneliti melakukan perbandingan pembagian data dengan 3 macam rasio. Perbandingan dari berbagai pembagian data dilakukan untuk mengetahui jumlah yang tepat untuk penelitian. Pembagian data dilakukan secara tidak *random* dan *random*. Dengan mencoba kedua hal tersebut diharapkan peneliti dapat mengetahui hasil yang lebih baik saat melakukan pembagian data. Hasil perbandingan pembagian data secara non-*random* dari K-Nearest Neighbors dan Decision Tree ditunjukan pada Tabel 5.3 dan Tabel 5.4.

Tabel 5.3 Perbandingan Pembagian Data K-NN

Rasio	**K-Optimal**	**RMSE**	**R^2**
90:10	7	0.0593	0.7750
80:20	7	0.0742	0.6615
70:30	7	0.0733	0.6494

Tabel 5.4 Perbandingan Pembagian Data Decision Tree

Rasio	**Max_depth**	**RMSE**	**R^2**
90:10	6	0.0628	0.8051
80:20	6	0.0787	0.6861
70:30	6	0.0765	0.6825

Berdasarkan hasil perbandingan pembagian data secara non-*random* yang sudah dilakukan, rasio 90:10 (train:test) untuk kedua algoritma memiliki nilai RMSE terendah dan R^2 tertinggi. Nilai RMSE rendah merupakan hasil yang baik karena menunjukan variasi nilai yang dihasilkan oleh model dekat dengan variasi nilai pada data training atau data pembelajaran yang dilakukan oleh mesin. Kemudian nilai R^2 yang tinggi menunjukan bahwa variabel dependen dan independen memiliki keterikatan sehingga merupakan hasil yang baik. Kemudian peneliti melakukan uji coba pembagian data secara *random* . Hasil perbandingan pembagian data secara *random* dengan ratio 90:10 untuk data *train:test* dari K-Nearest Neighbors dan Decision Tree ditunjukan pada Tabel 5.5

Tabel 5.5 Hasil Pembagian Data secara *Random*

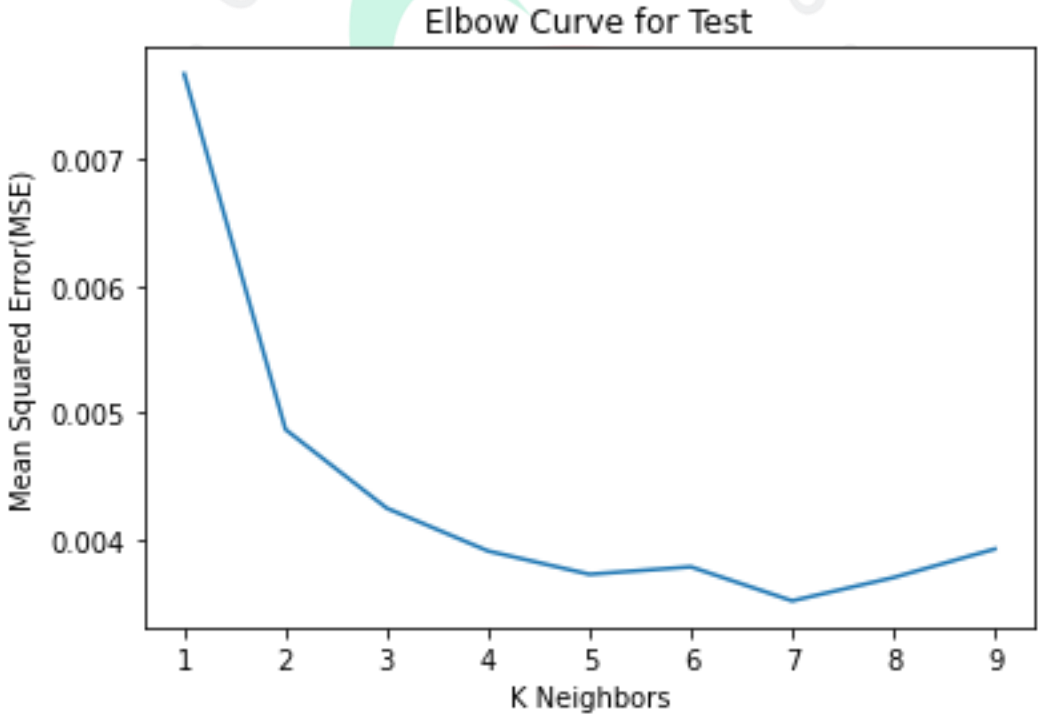
	RMSE	R-Square	Akurasi
7 (K-Optimal)	0.0808	0.4503	66.01%
6 (Max_Depth)	0.0835	0.3782	63.67%

Dapat disimpulkan bahwa pembagian data secara *random* memiliki hasil akurasi yang lebih rendah dibandingkan dengan non-*random* dengan rasio pembagian 90:10. Oleh karena itu, peneliti menggunakan rasio 90:10 dengan data *non random* untuk kedua pemodelan dengan algoritma K-Nearest Neighbors dan

Decision Tree. Semakin banyak data yang digunakan untuk pembelajaran atau *training* semakin banyak kelas yang terdapat pada data *training* sehingga titik uji akan semakin mudah menemukan variasi nilai yang mendekati di data *train*. Keseluruhan jumlah data 781 terbagi menjadi data *train* sebanyak 702 dan data *test* sebanyak 79.

5.1.3 Pemodelan dengan K-Nearest Neighbors

Pembuatan model perkiraan harga mobil bekas dengan algoritma K-Nearest Neighbors menggunakan *library* di Python yaitu *Sckit-Learn*. Langkah pertama yang dilakukan sebelum pembuatan model adalah menentukan K-optimal. Dengan menghitung jarak antar tetangga atau dikenal jarak Euclidean menggunakan metode Elbow. Hasil mencari K-optimal menggunakan metode Elbow ditunjukan pada Gambar 5.1.



Gambar 5.1 Metode Elbow (K-Optimal)

Berdasarkan grafik yang di atas, jumlah K yang memiliki nilai *mean squared error* (MSE) rendah merupakan hasil K-optimal. Model menjadi lebih baik dan memiliki tingkat akurasi yang tinggi apabila menggunakan nilai K-optimal.

Oleh karena itu, 7 ditetapkan sebagai K-optimal berdasarkan pencarian menggunakan metode Elbow. Kemudian pemodelan dilakukan dengan hasil yang ditunjukan pada Tabel 5.6.

Tabel 5.6 Hasil Pemodelan K-NN

K-Optimal	RMSE	R^2	Akurasi
7	0.0593	0.7750	82.61%

Hasil akurasi yang didapatkan dengan K-optimal dalam pemodelan harga mobil bekas menggunakan K-Nearest Neighbors mencapai 82,61%. Pemodelan ini dilakukan di luar dari variabel kondisi mobil bekas yang akan dijual, sehingga kondisi mobil dianggap sama rata dalam pemodelan. RMSE yang didapatkan melebihi ambang batas sebesar 0.0093 . Sementara itu, hasil R-*Square* dapat dikatakan akurat karena melebihi 0.7. Peneliti membuat tabel perbandingan supaya dapat melihat hasil akurasi apabila tidak menggunakan K-optimal. Perbandingan tersebut ditunjukan pada Tabel 5.7.

Tabel 5.7 Perbandingan K dan Hasil Akurasi

K	RMSE	R^2	Akurasi
1	0.0876	0.6475	62.03 %
2	0.0697	0.7460	75.92 %
3	0.0651	0.7700	78.99 %
4	0.0625	0.7784	80.67 %
5	0.0610	0.7782	81.57 %
6	0.0615	0.7670	81.28 %
7	0.0593	0.7750	82.61 %
8	0.0608	0.7543	81.70 %
9	0.0626	0.7288	80.58 %
10	0.0624	0.7253	80.74 %

Berdasarkan tabel perbandingan di atas, terbukti nilai K yang mempunyai akurasi paling tinggi merupakan K-optimal yang didapatkan melalui metode Elbow. Nilai RMSE dari K sama dengan tujuh paling rendah daripada nilai K lainnya. Hal tersebut membuat akurasi yang didapat lebih besar dari yang lain. Oleh karena itu, K sama dengan tujuh dipilih menjadi K di dalam pemodelan algoritma K-Nearest Neighbors.

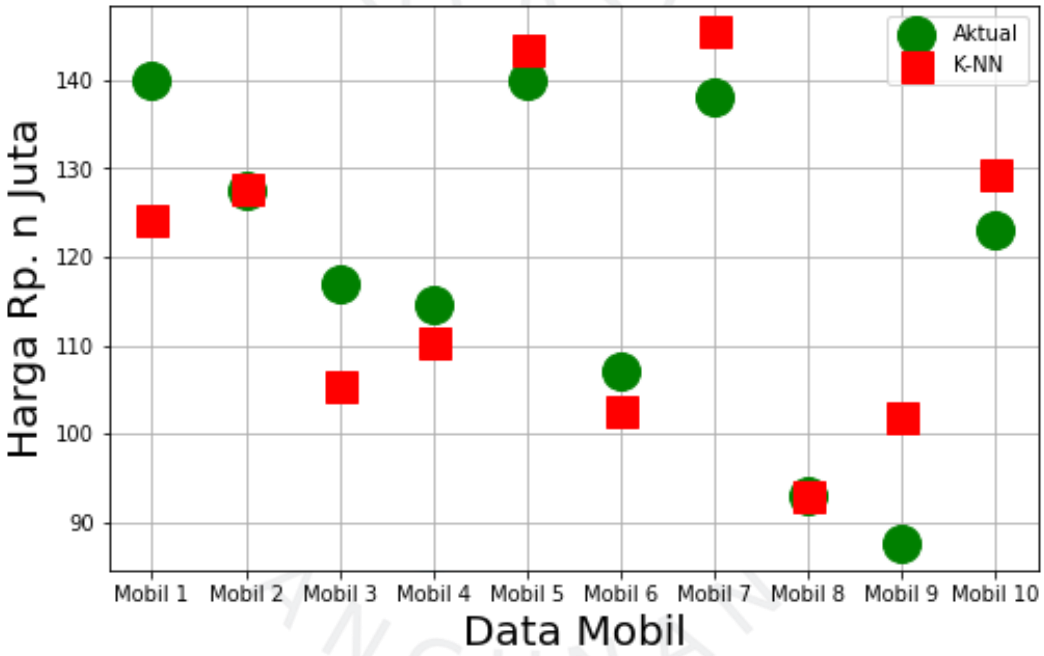
Kemudian peneliti melakukan pengujian harga mobil bekas menggunakan model yang sudah dibuat. Hal tersebut dilakukan dengan *input* data test ke dalam model. Data yang digunakan untuk *input* adalah model, umur, jarak satu, jarak dua, dan transmisi. Hasil prediksi yang dihasilkan oleh model K-Nearest Neighbors akan dibandingkan dengan harga aktual. Peneliti melakukan pengujian terhadap spesifik data dalam bentuk data hasil normalisasi. Kemudian untuk memudahkan pembacaan data tersebut dilakukan denormalisasi supaya kembali seperti data yang sebenarnya. Hasil prediksi dari pemodelan K-Nearest Neighbors tersebut dibandingkan dengan harga aktual. Hasil perbandingan harga prediksi dan aktual tersebut ditunjukan pada Tabel 5.8.

Tabel 5.8 Hasil Perbandingan Harga Prediksi dan Aktual - KNN

No	Model	Umur	Jarak Satu	Jarak Dua	Transmisi	Harga Prediksi	Harga Aktual
M1	Xenia	3	55000	60000.0	Manual	Rp. 124.14 JT	Rp. 140 JT
M2	Ertiga	5	50000	55000.0	*Automatic*	Rp. 127.69 JT	Rp. 127.5 JT
M3	Grand livina	6	115000	120000.0	Manual	Rp. 105.55 JT	Rp. 117 JT
M4	Avanza	7	100000	105000.0	Manual	Rp. 110.28 JT	Rp. 114.5 JT
M5	Grand livina	3	55000	60000.0	*Automatic*	Rp. 143.42 JT	Rp. 140 JT
M6	Xenia	6	115000	120000.0	Manual	Rp. 102.57 JT	Rp. 107 JT
M7	Xenia	2	15000	20000.0	Manual	Rp. 145.53 JT	Rp. 138 JT
M8	Avanza	11	150000	155000.0	Manual	Rp. 93 JT	Rp. 93 JT
M9	Grand livina	7	50000	55000.0	*Automatic*	Rp. 101.92 JT	Rp. 87.5 JT
M10	Grand livina	4	35000	40000.0	*Automatic*	Rp. 129.34 JT	Rp. 123 JT

Peneliti mengambil sepuluh data test untuk dijadikan pengujian hasil prediksi dari model yang sudah dibuat dari K-Nearest Neighbors. Sepuluh data tersebut sudah mencakup seluruh model mobil yang terdapat pada dataset. Data uji yang digunakan merupaka sepuluh *index* pertama pada pembagian data yang dilakukan oleh *machine learning*. Dengan data tersebut dapat dilihat hasil harga prediksi yang didapatkan dari pemodelan.

Peneliti membuat grafik dari data menggunakan perpustakaan dari Python yaitu matplotlib. Grafik untuk melihat perbandingan harga aktual dengan harga prediksi yang sudah dihasilkan dalam pemodelan harga mobil bekas. Grafik perbandingan harga aktual dan prediksi ditunjukan pada Gambar 5.2.



Gambar 5.2 Grafik Perbandingan Harga Aktual dan K-NN

Pengujian harga prediksi mobil bekas dengan K-Nearest Neighbors menghasil harga yang cukup mendekati harga aktual sehingga dapat dikatakan baik. Contoh prediksi terburuk pada kasus mobil 9 dimana kesalahan prediksi model adalah Rp. 14.42 JT atau 16.48%. Prediksi terburuk ini dapat diterima karena akurasi model adalah 82.61% sehingga kesalahan prediksi hingga 17.39% masih dapat ditoleransi.

5.1.4 Pemodelan dengan Decision Tree

Pembuatan model perkiraan harga mobil bekas dengan algoritma Decision Tree menggunakan *library* di Python yaitu *Sckit-Learn*. Peneliti melakukan pemodelan dengan cara melatih data menggunakan nilai max_depth yang merupakan maksimal kedalaman dari pohon. Langkah pertama yang dilakukan sebelum melakukan prediksi adalah mencari nilai max_depth terbaik. Pemodelan akan kurang tepat apabila kedalaman rendah dan tidak mencari nilai terbaik. Tidak ada satu nilai yang cocok untuk semua solusi (Mithrakumar, 2019). Oleh karena itu, mencari nilai max_depth diperlukan sebelum melakukan prediksi. Dengan percobaan nilai 1-10 untuk melakukan evaluasi hasil RMSE dan R^2. Hasil perbandingan nilai max_dept ditunjukan pada Tabel 5.9.

Tabel 5.9 Perbandingan Nilai Max_depth

Nilai Max_depth	RMSE	R^2
1	0.0987	0.0717
2	0.0724	0.6753
3	0.0738	0.6809
4	0.0721	0.7202
5	0.0663	0.7783
6	0.0628	0.8051
7	0.0693	0.7706
8	0.0709	0.7603
9	0.0717	0.7573
10	0.0763	0.7299

Berdasarkan tabel perbandingan di atas, nilai max_depth yang mempunyai nilai RMSE paling rendah dan R^2 paling tinggi adalah enam. Nilai max_depth lainnya memiliki RMSE lebih tinggi dan R^2 lebih rendah. Hal tersebut mempengaruhi hasil pemodelan. Oleh karena itu, max_depth sama dengan enam dipilih di dalam pemodelan algoritma Decision Tree. Hasil akurasi dari pemodelan yang dilakukan ditunjukan pada Tabel 5.10.

Tabel 5.10 Hasil Pemodelan Decision Tree

Nilai Max_depth	RMSE	R^2	Akurasi
6	0.0628	0.8051	80.49%

Hasil akurasi yang didapatkan dengan max_depth dalam pemodelan harga mobil bekas menggunakan Decision Tree mencapai 80,49%. Pemodelan ini dilakukan di luar dari variabel kondisi mobil bekas yang akan dijual, sehingga kondisi mobil dianggap sama rata dalam pemodelan. RMSE yang didapatkan melebihi ambang batas sebesar 0.0128 . Sementara itu, hasil R-*Square* dapat dikatakan akurat karena melebihi 0.7.

Kemudian peneliti melakukan pengujian harga mobil bekas menggunakan model yang sudah dibuat. Hal tersebut dilakukan dengan *input* data test ke dalam model. Data yang digunakan untuk *input* adalah model, umur, jarak satu, jarak dua, dan transmisi. Hasil prediksi yang dihasilkan oleh model Decision Tree akan dibandingkan dengan harga aktual. Hasil prediksi dalam bentuk normalisasi kemudian didenormalisasi supaya kembali seperti data yang sebenarnya. Kemudian hasil prediksi dari pemodelan Decision Tree tersebut dibandingkan dengan harga aktual. Hasil perbandingan harga prediksi dan aktual tersebut ditunjukan pada Tabel 5.11.

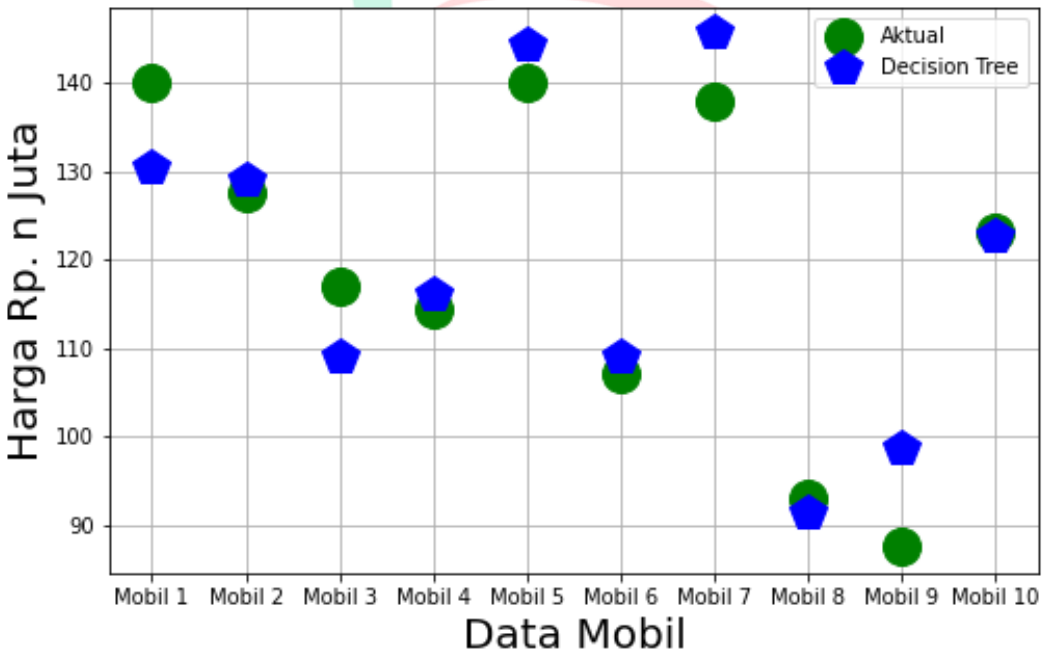
Tabel 5.11 Perbandingan Harga Prediksi dan Aktual - Decision Tree

No	Model	Umur	Jarak Satu	Jarak Dua	Transmisi	Harga Prediksi	Harga Aktual
M1	Xenia	3	55000	60000.0	Manual	Rp. 130.35 JT	Rp. 140 JT
M2	Ertiga	5	50000	55000.0	*Automatic*	Rp. 128.91 JT	Rp. 127.5 JT
M3	Grand livina	6	115000	120000.0	Manual	Rp. 109.09 JT	Rp. 117 JT
M4	Avanza	7	100000	105000.0	Manual	Rp. 116.14 JT	Rp. 114.5 JT
M5	Grand livina	3	55000	60000.0	*Automatic*	Rp. 144.15 JT	Rp. 140 JT
M6	Xenia	6	115000	120000.0	Manual	Rp. 109.09 JT	Rp. 107 JT
M7	Xenia	2	15000	20000.0	Manual	Rp. 145.56 JT	Rp. 138 JT
M8	Avanza	11	150000	155000.0	Manual	Rp. 91.25 JT	Rp. 93 JT

M9	Grand livina	7	50000	55000.0	*Automatic*	Rp. 98.63 JT	Rp. 87.5 JT
M10	Grand livina	4	35000	40000.0	*Automatic*	Rp. 122.63 JT	Rp. 123 JT

Peneliti mengambil sepuluh data test untuk dijadikan pengujian hasil prediksi dari model yang sudah dibuat dari Decision Tree. Sepuluh data tersebut sudah mencakup seluruh model mobil yang terdapat pada dataset. Data uji yang digunakan merupaka sepuluh *index* pertama pada pembagian data yang dilakukan oleh *machine learning*. Dengan data tersebut dapat dilihat hasil harga prediksi yang didapatkan dari pemodelan.

Kemudian peneliti membuat grafik menggunakan Python (matplotlib) untuk melihat perbandingan harga aktual dengan harga perkiraan yang sudah dihasilkan dalam pemodelan harga mobil bekas. Grafik perbandingan harga aktual dan prediksi ditunjukan pada Gambar 5.3.



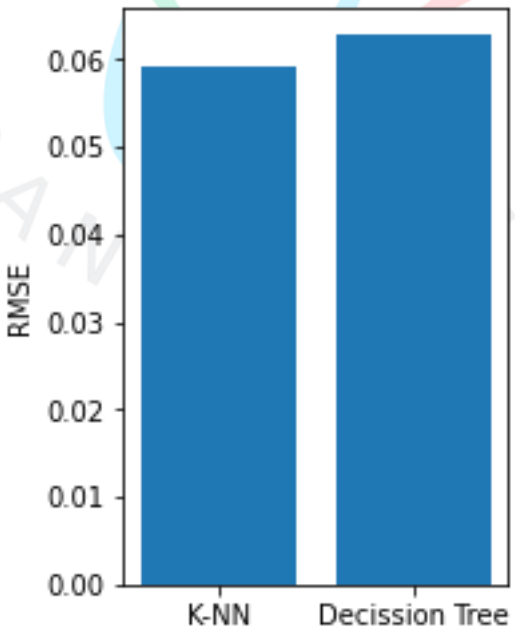
Gambar 5.3 Grafik Perbandingan Harga Aktual dan Decision Tree

Pengujian harga prediksi mobil bekas dengan Decision Tree menghasil harga yang cukup mendekati harga aktual sehingga dapat dikatakan baik. Contoh prediksi terburuk pada kasus mobil 9 dimana kesalahan prediksi model adalah Rp.

11.13 JT atau 12.72%. Prediksi terburuk ini dapat diterima karena akurasi model adalah 80.49% sehingga kesalahan prediksi hingga 19.51% masih dapat ditoleransi.

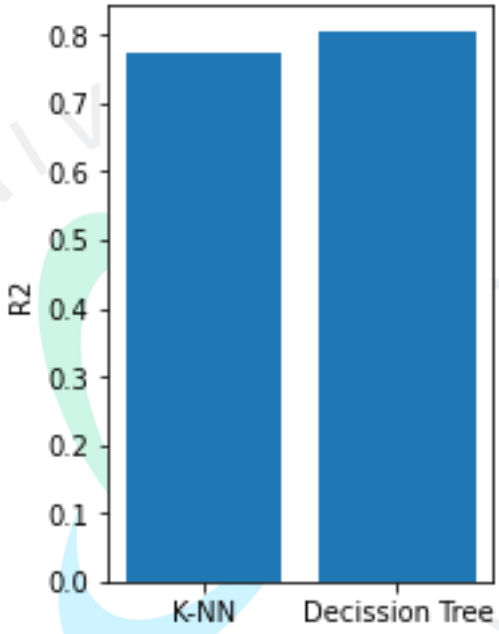
5.1.5 Perbandingan K-Nearest Neighbors dan Decision Tree

Setelah melakukan pemodelan dari algoritma K-Nearest Neighbors dan Decision Tree, peneliti membuat perbandingan dari kedua pemodelan yang sudah dibangun. Perbandingan pertama dari kedua algoritma dengan melihat nilai *Root Mean Square Error* (RMSE). Di dalam perbandingan tersebut Decision Tree memiliki nilai RMSE sebesar 0.0628, lebih tinggi dibandingkan dengan K-Nearest Neighbors yang memiliki nilai RMSE sebesar 0.0593 . Kedua algoritma mempunyai nilai RMSE melebihi dari ambang batas yaitu 0.05. Oleh karena itu, dilihat dari parameter RMSE pemodelan yang dilakukan dengan algoritma K-Nearest Neighbors sedikit lebih baik karena nilai RMSE yang lebih rendah menunjukan variasi nilai yang dihasilkan dalam pemodelan dekat dengan variasi nilai pada data pembelajaran atau pengamatan. Diagram batang dari perbandingan tersebut ditunjukan pada Gambar 5.4.



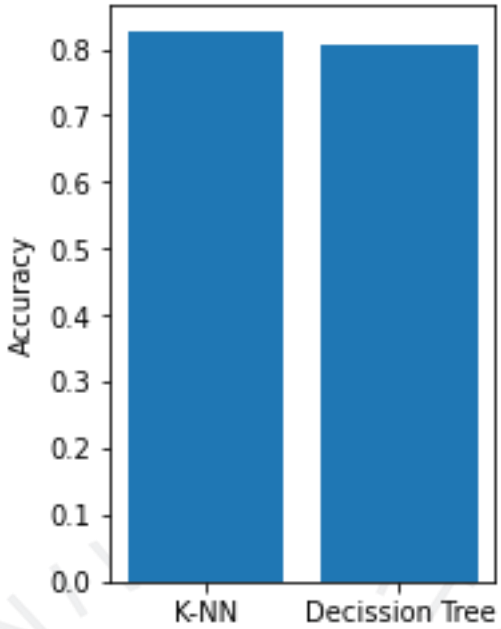
Gambar 5.4 Perbandingan K-NN dan Decision Tree (RMSE)

Perbandingan kedua yang dilakukan merupakan perbandingan R-*Square* dari kedua algoritma. Jika nilai R-*Square* semakin mendekati 1 maka semakin baik karena menunjukan besaran pengaruh dari dua atau lebih banyak fitur atau variabel bebas terhadap fitur yang diuji. Di dalam perbandingan tersebut Decision Tree memiliki nilai R-*Square* sebesar 0.8051, lebih tinggi dibandingkan dengan K-Nearest Neighbors yang memiliki nilai R-*Square* sebesar 0.7750 . Kedua algoritma mempunyai nilai R-*Square* lebih dari 0.7 dan dapat dikatakan model akurat. Diagram batang dari perbandingan tersebut ditunjukan pada Gambar 5.5.



Gambar 5.5 Perbandingan K-NN dan Decision Tree (R-*Square*)

Perbandingan ketiga yang dilakukan merupakan perbandingan akurasi dari kedua algoritma. Jika nilai RMSE yang dihasilkan semakin rendah maka pemodelan akan menghasilkan akurasi yang tinggi. Pemodelan yang dilakukan oleh K-Nearest Neighbors mendapatkan hasil akurasi mencapai 82,61%. Sementara itu, pemodelan yang dilakukan oleh Decision Tree mendapatkan hasil akurasi mencapai 80,49%. Apabila keduanya dibandingkan maka pemodelan yang dilakukan dengan K-Nearest Neighbors mendapatkan hasil akurasi yang lebih tinggi dibandingkan dengan pemodelan yang dilakukan dengan Decision Tree. Diagram batang dari perbandingan akurasi ditunjukan pada Gambar 5.6.



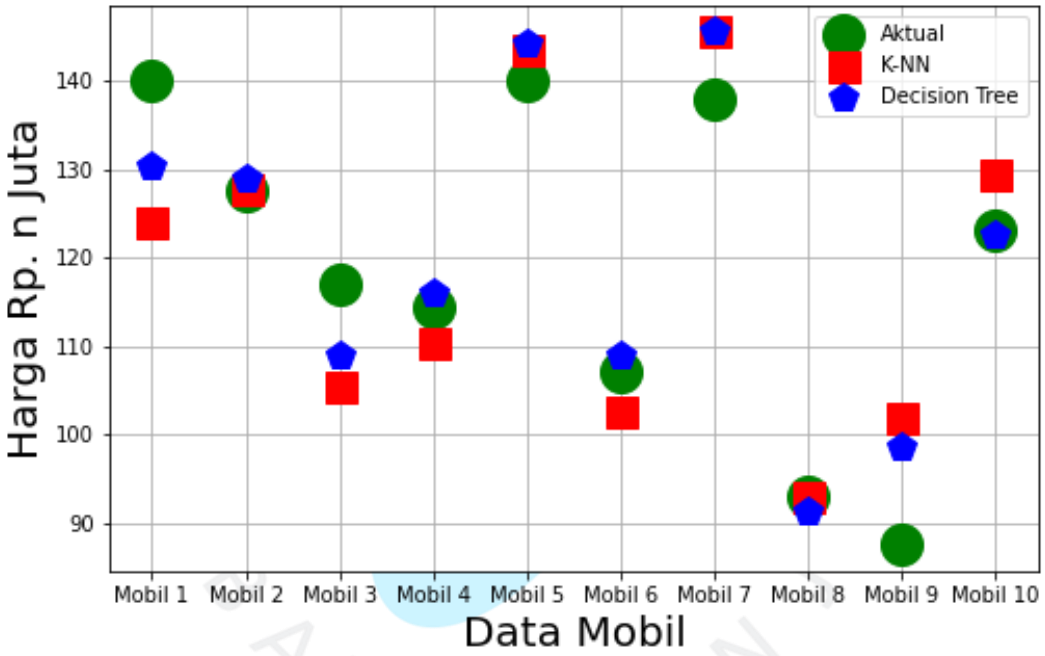
Gambar 5.6 Perbandingan K-NN dan Decision Tree (Akurasi)

Perbandingan ketiga yang dilakukan merupakan perbandingan harga prediksi dari kedua algoritma dengan harga aktual. Perbandingan harga prediksi dan aktual ditunjukan pada tabel 5.12.

Tabel 5.12 Tabel Perbandingan Harga Aktual, KNN, dan Decision Tree

No	Harga Prediksi (KNN)	Harga Prediksi (Decision Tree)	Harga Aktual
M1	Rp. 124,14 JT	Rp. 130,35 JT	Rp. 140 JT
M2	Rp. 127,69 JT	Rp. 128,91 JT	Rp. 127,5 JT
M3	Rp. 105,55 JT	Rp. 109,09 JT	Rp. 117 JT
M4	Rp. 110,28 JT	Rp. 116,14 JT	Rp. 114,5 JT
M5	Rp. 143,42 JT	Rp. 144,15 JT	Rp. 140 JT
M6	Rp. 102,57 JT	Rp. 109,09 JT	Rp. 107 JT
M7	Rp. 145,53 JT	Rp. 145,56 JT	Rp. 138 JT
M8	Rp. 93 JT	Rp. 91,25 JT	Rp. 93 JT
M9	Rp. 101.92 JT	Rp. 98.63 JT	Rp. 87.5 JT
M10	Rp. 129.34 JT	Rp. 122.63 JT	Rp. 123 JT

Peneliti membuat grafik untuk melihat perbandingan harga aktual dengan harga perkiraan yang sudah dihasilkan dalam pemodelan harga mobil bekas dengan kedua algoritma. Dari grafik tersebut kita dapat melihat perbedaan dari hasil yang sudah didapat melalui prediksi dari pemodelan dengan harga aktual dari data yang sebenarnya. Hasil pemodelan harga mobil bekas yang dilakukan berdasarkan variabel yang sudah ditetapkan pada batasan penelitian sudah mencapai akurasi yang cukup baik. Hasil dari grafik perbandingan harga aktual dan prediksi dari kedua algoritma ditunjukan pada gambar 5.7.



Gambar 5.7 Perbandingan Harga Aktual, K-NN dan Decision Tree

Beberapa hasil pemodelan dari kedua algoritma memiliki perbedaan harga yang cukup jauh dari harga yang sebenarnya. Pemodelan harga mobil bekas dilakukan dengan fitur-fitur yang mempengaruhi hasil dari harga perkiraan. Fitur-fitur tersebut adalah model mobil, umur, jarak tempuh, transmisi, dan kapasitas mesin. Fitur-fitur lain yang tidak digunakan oleh peneliti mempunyai peluang untuk mempengaruhi pemodelan. Contohnya, harga aktual lebih rendah dari harga prediksi yang dihasilkan pemodelan. Hal tersebut mempunyai kemungkinan seperti kondisi mesin mobil, kondisi badan mobil (mulus atau lecet), dan lainnya yang tidak dijadikan fitur di dalam pemodelan yang dilakukan peneliti. Oleh karena itu,

peneliti melakukan variasi data dengan melakukan uji coba penggunaan variabel input. Hal tersebut bertujuan untuk melihat input yang memiliki hasil lebih baik. Hasil uji coba variasi data tersebut ditunjukan pada Tabel.5.13.

Tabel 5.13 Uji Coba Variabel Input

No	Input	K-Optimal	Max Depth	Akurasi (K-NN)	Akurasi (DT)
1	Model, Umur, Jarak Satu, Jarak Dua, Transmisi	7	6	82.61%	80.49%
2	Model, Umur, Jarak Satu, Jarak Dua	9	5	79.32%	77.18%
3	Model, Umur, Transmisi	4	5	81.23%	81.26%
4	Model, Jarak Satu, Jarak Dua, Transmisi	9	5	45.25%	46.39%
5	Umur, Jarak Satu, Jarak Dua, Transmisi	8	5	79.87%	80.90%

Dari Tabel 5.12 dapat disimpulkan bahwa pemodelan harga mobil bekas dengan input model, umur, jarak satu, jarak dua, dan transmisi memiliki nilai akurasi yang lebih tinggi dari uji coba input yang lain. Oleh karena itu, peneliti memutuskan untuk tetap menggunakan variabel input tersebut sebagai masukan yang mempengaruhi harga prediksi mobil bekas.

5.2 Hasil dan Pembahasan Pengujian *White Box*

Pengujian terhadap kode program dengan melihat apakah masukan dan keluaran sudah sesuai dengan spesifikasi yang butuhkan atau belum menggunakan metode *White Box*. Pengujian tersebut ditunjukan pada tabel 5.14.

Tabel 5.14 Pengujian Kode Program

No	Kode Program	Hasil yang Diharapkan	Hasil Pengujian

1.	`import pandas as pd`
	`import numpy as np`
	`import matplotlib.pyplot as plt`
	`import sklearn.metrics as metrics`
		Melakukan import library yang akan digunakan.	Valid

2.	`data_location ='/content/drive/MyDrive/Data_AI`
	`raw_data = pd.read_csv(data_location)`
	`raw_data.shape`
		Data asli dimasukan ke dalam variabel, dan melihat ukuran data asli.	Valid

3.	`data = raw_data.dropna(axis=0)` `data.shape`	Data yang tidak lengkap dibersihkan, dan melihat ukuran data.	Valid
4.	`data_test_1 = data.iloc[:79,:]` `data_train_1 = data.iloc[79:,:]`	Pembagian lokasi data *test* dan *train* pertama.	Valid

5.	`#Normalisasi`
	`from sklearn.preprocessing import MinMaxScaler`
	`from sklearn import preprocessing`
	
	`scaler = preprocessing.MinMaxScaler()`
	`names = data.columns`
	`d = scaler.fit_transform(data)`
	`scaled_df = pd.DataFrame(d, columns=names)`
	`scaled_df.head()`
		Melakukan normalisasi data dan memasukan data hasil normalisasi ke variabel baru.	Valid

6.	`features = ['Model','Umur','Jarak Satu','Jarak Dua','Transmisi']`
	`x= scaled_df[features]`
	`y= scaled_df['Harga']`
	`x.shape, y.shape`
		Pemilihan fitur yang digunakan, menetapkan fitur yang ingin diuji serta fitur yang akan	Valid

		mempengaruhi hasil pengujian, dan melihat ukuran x dan y.	

7.	`from sklearn.model_selection import train_test_split`
	`train_x, test_x, train_y, test_y = train_test_split(x, y,`
	`test_size=0.1,`
	`random_state=0,`
	`shuffle=False)`
		Membagi data *train* dan data *test* sebesar 90:10.	Valid

8.	`data_test = scaled_df.iloc[:79,:]`
	`data_train = scaled_df.iloc[79:,:]`
	
	`print("Data Train :", train_x.shape)`
	`print("Data Test :", test_x.shape)`
		Menentukan lokasi data test dan train setelah dinormalisasi, dan melakukan cetak ukuran data train dan data test.	Valid

9.	`from sklearn.neighbors import KNeighborsRegressor as KNN_Reg`
	`from sklearn.tree import DecisionTreeRegressor as DeTree`
	`from sklearn.metrics import mean_squared_error as mse`
	`from sklearn.metrics import mean_squared_error`
	`from sklearn.metrics import r2_score`
	`from math import sqrt`
		Import dari *library* scikit-learn untuk kebutuhan pemodelan algoritma.	Valid

10.

```
def Elbow(K):
  test_mse =[]

  for i in K:
    model = KNN_Reg(n_neighbors=i)
    model.fit(train_x,train_y)
    tmp = model.predict(test_x)
    tmp = mse(tmp,test_y)
    test_mse.append(tmp)

  return test_mse

K = range(1,10)
test = Elbow(K)

plt.plot(K,test)
plt.xlabel('K Neighbors')
plt.ylabel('Mean Squared Error(MSE)')
plt.title('Elbow Curve for Test')
```

Mencari K-optimal menggunakan metode Elbow

Valid

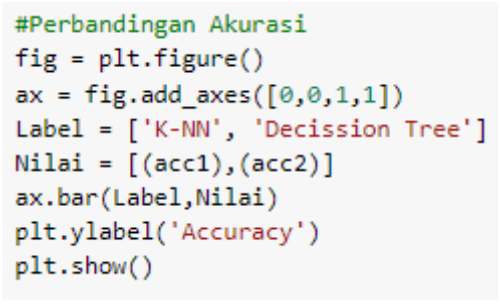
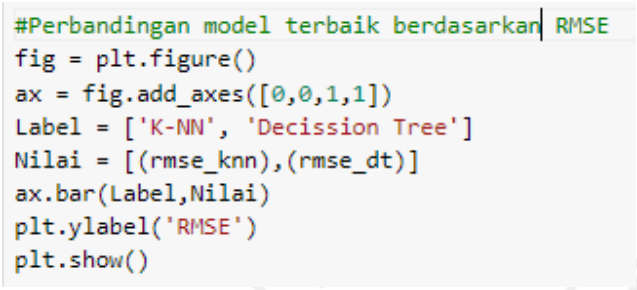
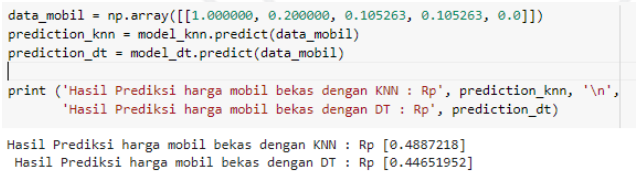
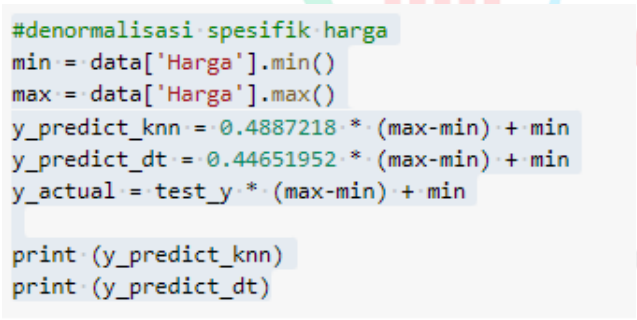
11.

```
#define model KNN
model_knn= KNN_Reg(n_neighbors = 7)
model_dt= DeTree(max_depth=6,random_state=0,)

# training the model KNN
model_knn.fit(train_x, train_y)
model_dt.fit(train_x, train_y)

#Akurasi
acc1 = model_knn.score(test_x,test_y)
acc2 = model_dt.score(test_x,test_y)

# test for prediction
test_predict_knn = model_knn.predict(test_x)
test_predict_dt = model_dt.predict(test_x)

#MSE
mse_knn = mse(test_predict_knn, test_y)
mse_dt = mse(test_predict_dt, test_y)

#RMSE
rmse_knn = sqrt(mean_squared_error(test_predict_knn,test_y))
rmse_dt = sqrt(mean_squared_error(test_predict_dt,test_y))

#R2
r2_knn =r2_score(test_predict_knn,test_y)
r2_dt =r2_score(test_predict_dt,test_y)
```

```
print('MSE K-NN: ', mse_knn,'\n', 'MSE DecisionTree: ', mse_dt,)
print('RMSE K-NN: ', rmse_knn,'\n','RMSE DecisionTree: ', rmse_dt,)
print('Skor R2 K-NN: ', r2_knn,'\n', 'Skor R2 DecisionTree: ', r2_dt,)
print('Accuracy KNN: ', acc1*100,'%','\n','Accuracy DecisionTree: ', acc2*100,'%')
```

Pemodelan K-NN dan Decision Tree, mencari akurasi, MSE, RMSE, R^2.

Valid

12.		Menampilkan diagram batang perbandingan akurasi antara kedua algoritma.	Valid
13.		Menampilkan diagram batang perbandingan RMSE antara kedua algoritma.	Valid
14.		Melakukan prediksi sesuai dengan data mobil yang dimasukkan ke dalam array.	Valid
15.		Denormalisasi spesifik harga hasil prediksi yang sudah dicari sebelumnya dan mencetaknya.	Valid