BAB III
HASIL DAN PEMBAHASAN

3.1 Data *Collection*

Data *Collection* merupakan langkah yang digunakan untuk menggali informasi dari data yang nantinya akan digunakan. Proses ini penting untuk memahami data secara lebih mendalam sebelum dilakukan tahap Data *Preprocessing*.

```
import pandas as pd
data = pd.read_csv('Pencetakan_KTP.csv')
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 350 entries, 0 to 349
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   KK                 350 non-null    int64
 1   NIK                350 non-null    int64
 2   Nama               350 non-null    object
 3   Umur               350 non-null    int64
 4   Jenis_Kelamin      350 non-null    object
 5   RT                 349 non-null    float64
 6   Kelurahan          349 non-null    object
 7   Keterangan         346 non-null    object
 8   Tanggal_Pengajuan  350 non-null    object
 9   Waktu_Pengajuan    350 non-null    object
 10  Waktu_Selesai      350 non-null    object
dtypes: float64(1), int64(3), object(7)
memory usage: 30.2+ KB
```

Gambar 3.1 Data Collection

Gambar 3.1 memberikan informasi tentang setiap kolom dalam dataset, termasuk jumlah nilai non-null, dan tipe data dari setiap kolom. Berdasarkan output tersebut, dataset terdiri dari total 11 kolom, 4 kolom dengan tipe data numerik dan 7 kolom dengan tipe data objek, dengan jumlah data sebanyak 350 data. Beberapa kolom memiliki nilai non-null yang kurang dari total baris dataset, menandakan adanya *Missing Value* dalam dataset tersebut.

3.2 Data *Preprocessing*

Pada bagian ini, akan dijelaskan hasil dari setiap langkah persiapan data yang telah dilakukan, termasuk pembersihan data, seleksi data, dan transformasi data. Proses ini sangat penting untuk memastikan bahwa data yang digunakan dalam pemodelan prediktif adalah data yang bersih, relevan, dan dalam format yang sesuai. Adapun penjelasan setiap langkah sebagai berikut:

3.2.1 Pembersihan Data

Pembersihan data adalah langkah awal yang sangat penting untuk memastikan bahwa dataset yang digunakan tidak mengandung nilai hilang yang dapat mempengaruhi akurasi model. Pada tahap ini, hal yang akan dilakukan dapat dilihat pada Gambar 3.2.

```
data.isnull().sum()
```

```
KK                  0
NIK                 0
Nama                0
Umur                0
Jenis_Kelamin       0
RT                  1
Kelurahan           1
Keterangan          4
Tanggal_Pengajuan   0
Waktu_Pengajuan     0
Waktu_Selesai       0
dtype: int64
```

```
data = data.dropna()
```

Gambar 3.2 Data Preprocessing

Perintah ***'data.isnull().sum()'*** pada Gambar 3.2 digunakan untuk menghitung jumlah nilai *Missing Value* di setiap kolom DataFrame. Kemudian, dengan menggunakan perintah ***'data.dropna()'***, baris-baris yang mengandung *Missing Value* dihapus, sehingga menghasilkan DataFrame baru yang telah dibersihkan dari *Missing Value*. Data yang awalnya berjumlah 350 data berubah menjadi 344 data setelah dilakukan pembersihan data

3.2.2 Seleksi Data

Seleksi data dilakukan untuk menghapus atribut yang tidak relevan sehingga hanya data yang signifikan dan relevan yang digunakan untuk membangun model prediksi. Hal ini dilakukan Regresi Linier Berganda memerlukan variabel independen yang bersifat numerik atau kategorikal yang dapat diukur.

```
data = data.drop(columns=['KK','NIK','Nama','Jenis_Kelamin','Kelurahan','Tanggal_Pengajuan'])
```

Gambar 3.3 Seleksi Data

Perintah ***'data.drop()'*** pada Gambar 3.3 digunakan untuk menghapus kolom-kolom tertentu dari DataFrame. Dalam hal ini, kolom yang dihapus adalah 'KK', 'NIK', 'Nama', dan 'Tanggal_Pengajuan'. Penggunaan parameter columns = ['KK', 'NIK', 'Nama', 'Jenis_Kelamin', 'Kelurahan' 'Tanggal_Pengajuan'] menginstruksikan Python untuk menghapus kolom-kolom tersebut dari DataFrame.

3.2.3 Transformasi Data

Transformasi data dilakukan untuk mengubah data kategorik menjadi bentuk numerik yang dapat diproses oleh model regresi linear. Proses ini menggunakan teknik *Label Encoding* dan melibatkan beberapa tahapan yang ditunjukan pada Gambar 3.4.

```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
data['Keterangan'] = le.fit_transform(data['Keterangan'])

data['Waktu_Pengajuan'] = data['Waktu_Pengajuan'].apply(lambda x: int(x.split(':')[0]) * 60 + int(x.split(':')[1]) - 480)
data['Waktu_Selesai'] = data['Waktu_Selesai'].apply(lambda x: int(x.split(':')[0]) * 60 + int(x.split(':')[1]) - 480)
```

Gambar 3.4 Transformasi Data

LabelEncoder dari *library scikit-learn* untuk mengkonversi nilai-nilai dalam kolom 'Keterangan' menjadi bilangan bulat yang unik. Ini dilakukan dengan memanggil Metode ***'fit_transform()'*** dari objek *LabelEncoder* pada setiap kolom tersebut. Adapun nilai kolom yang berubah dapat dilihat pada Tabel 3.1 berikut:

Tabel 3.1 Label Encoding Keterangan

Keterangan	Label Encoding
Hilang	0
PRR / Baru Rekam	1
Pemekaran	2
Pindahan	3
Rusak	4
Ubah Data	5

Selain itu, format waktu pada kolom 'Waktu_Pengajuan' dan 'Waktu_Selesai' di ubah dari jam menjadi menit menggunakan fungsi ***'lambda'*** yang memecah string waktu dan mengonversi jam dan menit ke dalam satuan menit. Proses ini penting dalam mempersiapkan data, terutama ketika menggunakan Metode *Machine Learning* yang memerlukan input dalam bentuk numerik. Tabel 3.2 adalah contoh konversi yang dilakukan.

Tabel 3.2 Konversi Waktu

Format Jam	Satuan Menit
8:05	5
8:23	32
9:19	79
9:47	107
10:10	130

Fungsi ***'apply'*** digunakan untuk menerapkan fungsi ***'lambda'*** pada setiap elemen dalam kolom. Kemudian fungsi 'lambda' akan memisahkan waktu ke dalam jam dan menit menggunakan ***'split(':')'***, mengkonversi jam menjadi menit dengan mengalikannya dengan 60, lalu menambahkan menitnya. Setelah itu hasilnya akan dikurangkan dengan 480 karena waktu yang diambil hanya dari pukul 8 pagi.

Hasil akhirnya adalah waktu dalam menit yang menggantikan nilai asli dalam kolom 'Waktu_Pengajuan' dan 'Waktu_Selesai'.

3.3 Implementasi Model

Langkah awal yang dilakukan dalam implementasi model adalah mengimpor data yang telah dikumpulkan sebelumnya dan menampilkannya dalam bentuk DataFrame. Data ini mencakup berbagai atribut yang relevan untuk prediksi waktu pencetakan KTP. Dengan menggunakan *Library* seperti *pandas*, data di *import* dan ditampilkan sehingga memudahkan dalam pengelolaan dan analisis lebih lanjut.

```
import pandas as pd
data = pd.read_csv('Pencetakan_KTP.csv')
```

Gambar 3.5 Import Data

Dengan menjalankan perintah ***'pd.read_csv(' ')'*** pada Gambar 3.5, data yang ada di file CSV akan dibaca dan dimuat ke dalam variabel data, sehingga dapat diolah dan dianalisis lebih lanjut menggunakan berbagai fungsi yang disediakan oleh *pandas*. Proses ini merupakan langkah awal yang penting dalam analisis data, memungkinkan data tersebut diakses dan diolah dengan mudah dalam penggunaan Python.

3.3.1 Data Correlation

Langkah selanjutnya adalah melakukan analisis korelasi untuk mengidentifikasi hubungan antara atribut dengan label. Tujuan dari analisis ini adalah untuk menemukan atribut yang memiliki pengaruh signifikan terhadap waktu pencetakan. Dengan menggunakan Metode korelasi, atribut-atribut dalam dataset dianalisis untuk menemukan nilai korelasi positif maupun negatif terbesar terhadap label.

```
korelasi = data.corr()
korelasi
```

Gambar 3.6 Korelasi Data

Fungsi ***'data.corr()'*** pada Gambar 3.6 digunakan untuk menghitung matriks korelasi dari semua pasangan kolom dalam DataFrame. Nilai korelasi berkisar antara -1 dan 1, dimana nilai 1 menunjukkan korelasi positif sempurna, nilai -1 menunjukkan korelasi negatif sempurna, dan nilai 0 menunjukkan tidak ada korelasi linear. Dengan menggunakan matriks korelasi ini, dapat diidentifikasi atribut-atribut mana yang memiliki hubungan paling kuat dengan label, sehingga atribut-atribut yang signifikan dapat dipilih untuk digunakan dalam model prediksi. Gambar 3.7 menunjukan nilai korelasi pada tiap atribut.

	Umur	RT	Keterangan	Waktu_Pengajuan	Waktu_Selesai
Umur	1.000000	0.043878	0.367494	-0.064426	-0.121579
RT	0.043878	1.000000	0.049949	-0.062125	-0.061679
Keterangan	0.367494	0.049949	1.000000	-0.155001	-0.203164
Waktu_Pengajuan	-0.064426	-0.062125	-0.155001	1.000000	0.989696
Waktu_Selesai	-0.121579	-0.061679	-0.203164	0.989696	1.000000

Gambar 3.7 Hasil Korelasi

Dalam pengembangan model prediksi waktu pencetakan KTP ini, tiga atribut yang dipilih sebagai yang paling berpengaruh terhadap label 'Waktu_Selesai', yaitu 'Umur', 'Keterangan', dan 'Waktu_Pengajuan'. Pengambilan keputusan ini didasarkan pada hasil analisis korelasi, di mana ketiga atribut ini memiliki korelasi yang cukup signifikan dengan 'Waktu_Selesai'. 'Umur' dan 'Keterangan' memiliki korelasi negatif yang lemah, sementara 'Waktu_Pengajuan' memiliki korelasi positif yang lebih kuat dengan 'Waktu_Selesai'.

3.3.2 Split Data

Setelah mengidentifikasi atribut yang signifikan, langkah selanjutnya adalah membagi dataset menjadi dua bagian, yaitu data *Training* dan data *Testing*. Untuk menentukan rasio yang optimal antara data *Training* dan data *Testing*, dilakukan pengujian terhadap beberapa rasio perbandingan untuk melihat rasio mana yang menghasilkan akurasi tertinggi.

Tabel 3.3 Rasio Perbandingan

Rasio Perbandingan	Akurasi Data Training	Akurasi Data Testing
70:30	0.985058	0.980566
80:20	0.984837	0.979068
90:10	0.985317	0.966859

Berdasarkan hasil pengujian yang ditampilkan pada tabel 3.3, rasio 70:30 menunjukkan akurasi tertinggi, dengan nilai akurasi sebesar 0,98 untuk data *Training* serta data *Testing*. Dari hasil ini, dapat disimpulkan bahwa rasio 70:30 memberikan akurasi terbaik untuk pembagian data *Training* dan *Testing*. Oleh karena itu, rasio ini digunakan dalam proses pelatihan model.

```
from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(x, y, train_size=0.7, random_state = 42)
```

Gambar 3.8 Split Data

Fungsi ***'train_test_split'*** dari *Library scikit-learn* pada Gambar 3.8 adalah untuk membagi dataset menjadi dua bagian. Variabel *x* berisi fitur atau atribut yang akan digunakan untuk prediksi, sedangkan

y berisi label atau target yang akan diprediksi. Dengan parameter ***'train_size=0.7'***, sebanyak 70% dari data akan dialokasikan untuk data training (*xtrain* dan *ytrain*) dan 10% sisanya untuk data *Testing* (*xtest* dan *ytest*). Penggunaan random_state=42 memastikan bahwa pembagian data dilakukan secara acak namun konsisten.

3.3.3 Modelling Regresi Linier Berganda

Dengan data yang telah dibagi, Metode Regresi Linier Berganda diterapkan untuk membangun model prediksi. Model dilatih menggunakan data training yang telah disiapkan sebelumnya. Metode Regresi Linier Berganda digunakan untuk memprediksi waktu pencetakan KTP berdasarkan atribut-atribut yang telah dipilih melalui analisis korelasi.

```
from sklearn.linear_model import LinearRegression
reg = LinearRegression().fit(xtrain, ytrain)

y_pred = reg.predict(xtest)
```

Gambar 3.9 Modelling Regresi Linier Berganda

Model *LinearRegression* dari *Library scikit-learn* pada gambar 3.9 digunakan untuk memprediksi nilai target berdasarkan data yang digunakan. Proses ini dimulai dengan melatih model Regresi Linier Berganda menggunakan data *Training* yang terdiri dari fitur (*x*) dan target (*y*) melalui Metode ***'.fit()'***. Setelah model dilatih, model tersebut kemudian digunakan untuk memprediksi nilai-nilai target pada data uji ($xtest$) dengan menggunakan Metode ***'.predict()'***. Hasil prediksi ini disimpan dalam ***'y_pred'***, yang berisi nilai-nilai target yang diprediksi berdasarkan model yang telah dilatih sebelumnya.

3.3.4 Hasil

Hasil prediksi model berbentuk satuan menit, oleh karena itu dibuat sebuah fungsi untuk mengkonversi hasil prediksi menjadi format jam. Fungsi ini akan mengubah satuan menit menjadi format waktu yang lebih terstruktur. Fungsi yang dibuat dapat dilihat pada gambar 3.10.

```
def convert_integer_to_time(minutes):
    jam = minutes // 60
    menit = minutes % 60
    return f"{jam:02d}:{menit:02d}"
```

Gambar 3.10 Konversi Hasil

Fungsi ini menghitung jumlah jam dengan membagi total menit dengan 60 menggunakan operator (//), yang menghasilkan jumlah jam. Selanjutnya, menggunakan operator (%) untuk menghitung sisa waktu dalam bentuk menit. Hasilnya diformat menggunakan *f-string* (***f"{jam:02d}:{menit:02d}"***), yang akan menampilkan waktu dalam format jam dan menit.

```
data_prediksi = pd.DataFrame({
    'Selesai': y_pred.astype(int) - xtest['Waktu_Pengajuan'].astype(int),
    'Waktu_Selesai': y_pred.astype(int)
})

data_prediksi['Selesai'] = data_prediksi['Selesai'].astype(str) + ' menit'
data_prediksi['Waktu_Selesai'] = data_prediksi['Waktu_Selesai'].apply(convert_integer_to_time)
```

Gambar 3.11 Menyimpan Hasil

Gambar 3.11 merupakan langkah terakhir yang dilakukan untuk membuat sebuah DataFrame baru ***'data_prediksi'*** menggunakan library pandas. DataFrame ini memiliki satu kolom awal yang disebut 'Waktu_Selesai', berisi nilai prediksi ***'y_pred'*** yang sudah diubah tipe datanya menjadi integer. Selanjutnya, kolom 'Waktu_Selesai' ditambahkan dengan menerapkan fungsi *convert_integer_to_time* ke setiap nilai dalam kolom. Hasil akhirnya berisi prediksi 'Waktu_Selesai' dalam format waktu yang lebih terstruktur dan mudah dipahami.

Tabel 3.4 Hasil Prediksi

No	Selesai	Waktu_Selesai
1	14 menit	09:36
2	12 menit	10:38
3	30 menit	11:45
4	36 menit	15:06
5	14 menit	08:31
…	…	…
100	14 menit	11:21
101	27 menit	10:08
102	5 menit	11:22
103	6 menit	10:30
104	22 menit	13:26

Tabel 3.4 menampilkan hasil prediksi berdasarkan model Regresi Linier Berganda, dimana kolom 'Selesai' adalah lama waktu pencetakan KTP dalam bentuk menit dan kolom 'Waktu_Selesai' adalah hasil prediksi yang telah diubah menjadi format jam.

3.4 Perhitungan Regresi Linier Berganda

Pada penelitian ini Regresi Linear yang digunakan adalah Regresi Linear Berganda. Regresi Linear Berganda merupakan teknik statistik yang digunakan untuk memodelkan hubungan antara satu variabel dependen *Y* dengan dua atau lebih variabel independen *X1, X2, X3, ..., Xn*. Perhitungan regresi linear berganda melibatkan beberapa langkah utama, yaitu membentuk matriks, menghitung determinan, dan menentukan nilai koefisien. Adapun data yang digunakan dapat dilihat pada tabel 3.5.

Tabel 3.5 Data Perhitungan Regresi Linear

No	Umur	Keterangan	Waktu_Pengajuan	Waktu_Selesai
1	56	3	423	435
2	23	5	352	360
3	18	3	156	169
4	39	3	22	32
5	17	1	403	462
…	…	…	…	…
236	18	1	17	49
237	78	0	125	138
238	17	1	91	109
239	17	1	342	370
240	23	5	75	84
Total	7034	546	40082	44873

Data yang digunakan adalah data bersih yang telah diolah pada tahap *Preporcessing*, dengan total data sebanyak 240 data *Training*. Untuk mendapatkan nilai yang nantinya akan digunakan dalam perhitungan matriks A, maka di buat tabel pembantu dengan melakukan perhitungan terhadap variabel independen X1 (Umur), X2 (Keterangan), X3 (Waktu_Pengajuan), dan variabel dependen Y (Waktu_Selesai). Tabel 3.6 menujukan perhitungan yang dilakukan terhadap variabel independen serta variabel dependen.

Tabel 3.6 Tabel Pembantu

No	X1*Y	X2*Y	X3*Y	X1*X2	X1*X3	X2*X3	X1^2	X2^2	X3^2
1	24360	1305	184005	168	23688	1269	3136	9	178929
2	8280	1800	126720	115	8096	1760	529	25	123904
3	3042	507	26364	54	2808	468	324	9	24336
4	1248	96	704	117	858	66	1521	9	484
5	7854	462	186186	17	6851	403	289	1	162409
…	…	…	…	…	…	…	…	…	…
236	882	49	833	18	306	17	324	1	289
237	10764	0	17250	0	9750	0	6084	0	15625
238	1853	109	9919	17	1547	91	289	1	8281
239	6290	370	126540	17	5814	342	289	1	116964
240	1932	420	6300	115	1725	375	529	25	5625
Total	1274971	88863	12489149	17931	1164511	81100	255624	1816	11498286

Langkah selanjutnya adalah menghitung nilai Determinan *A*, *A0*, *A1*, *A2* dan *A3* dengan menggunakan matriks *A*. Karena terdapat 3 persamaan dengan 3 variabel yang tidak diketahui nilainya, yaitu *a*, *b1*, *b2* serta *b3*, persamaan tersebut dapat dinyatakan dalam persamaan matriks sebagai berikut:

$$\begin{bmatrix} n & \sum X_1 & \sum X_2 & \sum X_3 \\ \sum X_1 & \sum X_1^2 & \sum X_1 X_2 & \sum X_1 X_3 \\ \sum X_2 & \sum X_1 X_2 & \sum X_2^2 & \sum X_2 X_3 \\ \sum X_3 & \sum X_1 X_3 & \sum X_2 X_3 & \sum X_3^2 \end{bmatrix} \begin{bmatrix} a \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} \sum Y \\ \sum X_1 Y \\ \sum X_2 Y \\ \sum X_3 Y \end{bmatrix} \quad (3.1)$$

Dimana:

n = Jumlah Data

a = Konstanta (intercept)

b_i = Koefisien Regresi

Matriks pertama adalah matriks A, Matriks kedua adalah nilai-nilai yang akan dicari dan Matriks ketiga adalah nilai *H* yang merupakan hasil dari perkalian antara nilai-nilai *Y*. Dengan menggunakan hasil dari tabel pembantu, Maka perhitungan determinan dari matriks *A* adalah sebagai berikut:

$$A = \begin{bmatrix} 240 & 7034 & 546 & 40082 \\ 7034 & 255624 & 17931 & 1164511 \\ 546 & 17931 & 1816 & 81100 \\ 40082 & 1164511 & 81100 & 11498286 \end{bmatrix} = 27316156751950700$$

Untuk menghitung determinannya, kita bisa menggunakan Metode ekspansi kofaktor atau Metode lain yang sesuai. Setelah mendapatkan hasil determinan matriks *A*, langkah selanjutnya adalah menghitung matriks *A0*, *A1*, *A2*, serta *A3*. Berikut adalah perhitungan dari setiap matriks:

1) Matriks *A0* dihitung dengan menggantikan kolom pertama dengan nilai *H*

$$A_0 = \begin{bmatrix} 44873 & 7034 & 546 & 40082 \\ 1274971 & 255624 & 17931 & 1164511 \\ 88863 & 17931 & 1816 & 81100 \\ 12489149 & 1164511 & 81100 & 11498286 \end{bmatrix} = 983270794940830000$$

2) Matriks *A1* dihitung dengan menggantikan kolom kedua dengan nilai *H*

$$A_1 = \begin{bmatrix} 240 & 44873 & 546 & 40082 \\ 7034 & 1274971 & 17931 & 1164511 \\ 546 & 88863 & 1816 & 81100 \\ 40082 & 12489149 & 81100 & 11498286 \end{bmatrix} = -12814013437205300$$

3) Matriks *A2* dihitung dengan menggantikan kolom ketiga dengan nilai *H*

$$A_2 = \begin{bmatrix} 240 & 7034 & 44873 & 40082 \\ 7034 & 255624 & 1274971 & 1164511 \\ 546 & 17931 & 88863 & 81100 \\ 40082 & 1164511 & 12489149 & 11498286 \end{bmatrix} = -91013916371081000$$

4) Matriks *A3* dihitung dengan menggantikan kolom keempat dengan nilai *H*

$$A_3 = \begin{bmatrix} 240 & 7034 & 546 & 44873 \\ 7034 & 255624 & 17931 & 1274971 \\ 546 & 17931 & 1816 & 88863 \\ 40082 & 1164511 & 81100 & 12489149 \end{bmatrix} = 28182233421495900$$

Setelah mendapatkan semua nilai determinan matriks *A*, maka diperoleh persamaan untuk menghitung nilai *a* serta *b* sebagai berikut:

$$a = \frac{Det\ A_0}{Det\ A} \quad b_1 = \frac{Det\ A_1}{Det\ A} \quad b_2 = \frac{Det\ A_2}{Det\ A} \quad b_3 = \frac{Det\ A_3}{Det\ A} \tag{3.2}$$

1) Menghitung nilai *a*

$$a = \frac{Det\ A_0}{Det\ A} = \frac{983270794940830000}{27316156751950700} = 35{,}9959$$

2) Menghitung nilai *b1*

$$b_1 = \frac{Det\ A_1}{Det\ A} = \frac{-12814013437205300}{27316156751950700} = -0{,}4691$$

3) Menghitung nilai *b2*

$$b_2 = \frac{Det\ A_2}{Det\ A} = \frac{-91013916371081000}{27316156751950700} = -3{,}3319$$

4) Menghitung nilai *b3*

$$b_3 = \frac{Det\ A_3}{Det\ A} = \frac{28182233421495900}{27316156751950700} = 1{,}0317$$

Setelah mendapatkan nilai a dan b, langkah selanjutnya adalah membuat model persamaan regresi linear berganda menggunakan persamaan 2.2. Berikut adalah model persamaan regresi linear berganda:

$$Y = a + b_1X_1 + b_2X_2 + b_3X_3$$
$$Y = 35{,}9959 - 0{,}4691 * X_1 - 3{,}3319 * X_2 + 1{,}0317 * X_3$$

Dengan menggunakan persamaan diatas, akan dilakukan perhitungan linear regresi berganda menggunakan data *Testing* yang diambil setelah proses *Split Data*. Data *Testing* yang digunakan dapat dilihat pada Tabel 3.7.

Tabel 3.7 Data Testing

No	Umur	Keterangan	Waktu_Pengajuan
1	30	3	82
2	39	3	146
3	18	1	195
4	18	1	390
5	25	3	17
…	…	…	…
100	37	3	187
101	18	1	101
102	58	3	197
103	63	5	156
104	27	3	304

Dengan perbandingan, 70:30 didapatkan data *Testing* sebanyak 104 data. Data ini akan dihitung menggunakan persamaan yang telah didapat sebelumnya untuk mendapatkan hasil nilai prediksi. Perhitungan dari setiap data dapat dilihat pada Tabel 3.8 berikut:

Tabel 3.8 Hasil Perhitungan

No	Perhitungan Regresi Linear	Hasil Prediksi
1	$Y = 35{,}9959 - 0{,}4691(30) - 3{,}3319(3) + 1{,}0317(82)$	96,5271875
2	$Y = 35{,}9959 - 0{,}4691(39) - 3{,}3319(3) + 1{,}0317(146)$	158,3344480
3	$Y = 35{,}9959 - 0{,}4691(18) - 3{,}3319(1) + 1{,}0317(195)$	225,4028709
4	$Y = 35{,}9959 - 0{,}4691(18) - 3{,}3319(1) + 1{,}0317(390)$	426,5854738
5	$Y = 35{,}9959 - 0{,}4691(25) - 3{,}3319(3) + 1{,}0317(17)$	31,8118206
…	…	…
100	$Y = 35{,}9959 - 0{,}4691(37) - 3{,}3319(3) + 1{,}0317(187)$	201,5725802
101	$Y = 35{,}9959 - 0{,}4691(18) - 3{,}3319(1) + 1{,}0317(101)$	128,4225392
102	$Y = 35{,}9959 - 0{,}4691(58) - 3{,}3319(3) + 1{,}0317(197)$	202,0385335
103	$Y = 35{,}9959 - 0{,}4691(63) - 3{,}3319(5) + 1{,}0317(156)$	150,7293584
104	$Y = 35{,}9959 - 0{,}4691(27) - 3{,}3319(3) + 1{,}0317(304)$	326,9731436

3.5 Evaluasi Hasil

Evaluasi model adalah tahap terakhir yang dilakukan untuk mengukur kinerja model prediksi. Mean Absolute Error (MAE) adalah Metode pengujian yang digunakan untuk mengevaluasi akurasi model prediksi ini. Nilai MAE didapat dengan menghitung rata-rata kesalahan absolut antara nilai aktual dan hasil prediksi. Tabel 3.9 menunjukan perbandingan antara nilai aktual dan nilai prediksi, serta kesalahan absolut untuk setiap hasil prediksi.

Tabel 3.9 Perbandingan Nilai Aktual dan Prediksi

No	Nilai Aktual	Nilai Prediksi	Absolute Error
1	89	96,5271875	7,5271875
2	162	158,3344480	3,6655520
3	314	225,4028709	88,5971291
4	422	426,5854738	4,5854738
5	35	31,8118206	3,1881794
…	…	…	…
100	203	201,5725802	1,4274198
101	122	128,4225392	6,4225392
102	201	202,0385335	1,0385335
103	165	150,7293584	14,2706416
104	313	326,9731436	13,9731436
Total	…	…	1233,3953612

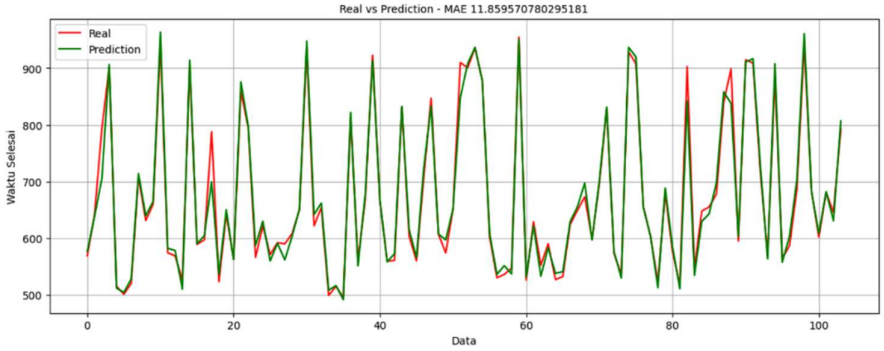
Nilai Absolute error dihitung sebagai perbedaan absolut antara nilai aktual dan nilai prediksi untuk setiap observasi. Dalam hal ini, total absolute error adalah 1233,3953612. Setelah itu, kita membagi total absolute error ini dengan jumlah data prediksi untuk mendapatkan nilai rata-rata dari kesalahan absolut. Karena ada 104 data prediksi dalam dataset ini, kita dapat menghitung MAE dengan persamaan berikut:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

$$MAE = \frac{1233,3953612}{104} = 11,85957078$$

Nilai MAE sebesar 11,85957078 menunjukkan bahwa, secara rata-rata prediksi model Regresi Linier Berganda meleset sekitar 12 menit dari nilai aktualnya. Nilai MAE yang lebih rendah mengindikasikan bahwa model memiliki kinerja yang lebih baik karena kesalahan prediksinya lebih kecil. Dalam konteks penelitian ini, nilai MAE membantu mengevaluasi sejauh mana model Regresi

Linier Berganda dapat diandalkan dalam memprediksi waktu pencetakan KTP berdasarkan atribut yang dipilih.



Gambar 3.12 Grafik Perbandingan

Perbandingan antara nilai aktual dan nilai prediksi juga dapat dilihat pada Gambar 3.12 di atas, dimana garis warna merah menunjukan data aktual sedangkan garis warna hijau menunjukan data hasil prediksi. Hal ini memberikan pandangan langsung tentang seberapa dekat atau jauh hasil nilai prediksi dari nilai sebenarnya, memungkinkan evaluasi yang lebih mendalam terhadap keakuratan model yang digunakan.

```
from sklearn.metrics import mean_absolute_error

mae=mean_absolute_error(ytrain,y_pred)
print(mae)
```

```
11.48745823630302
```

Gambar 3.13 Hasil MAE Data *Training*

Gambar 3.13 menunjukan bahwa pengujian nilai MAE juga dilakukan pada data *Training* yang menghasilkan nilai MAE sebesar 11,48745823, menunjukan bahwa model Regresi Linier Berganda memiliki kinerja yang cukup konsisten. Perbedaan kecil antara nilai MAE pada data *Training* dan *Testing* menandakan model tidak mengalami overfitting yang signifikan, sehingga dapat memprediksi data baru dengan kesalahan prediksi tidak jauh berbeda dari data pelatihan. Evaluasi ini menunjukkan bahwa model memiliki kinerja yang baik dan dapat diandalkan dalam memprediksi waktu pencetakan KTP berdasarkan atribut yang dipilih.

```
reg.score(x, y)
```

```
0.9838423659127112
```

Gambar 3.14 R^2 Score

Adapun hasil akurasi pada Gambar 3.14 yang dihasilkan dari model yang sudah diuji dengan data prediksi dalam bentuk R^2 yaitu sebesar 0.9838. Hasil tersebut mengindikasikan bahwa hubungan linear

yang kuat antara atribut Umur, Keterangan, serta Waktu Pengajuan dengan label Waktu Selesai. Semakin nilai R^2 mendekati satu, maka semakin baik prediksi yang dihasilkan oleh model prediksi Regresi Linier Berganda.