Nama: Andi Cleopatra Maryam Jamila

Nim: 1103213071

Regression model

Memuat Library

```
# Import Library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
from sklearn.model_selection import train_test_split, GridSearchCV, RandomizedSearchCV
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor
from xgboost import XGBRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

Memasukkan beberapa library yang dibutuhkan dalam menyelesaikan regression model.

Memuat Dataset

```
[ ] # Load dataset
    df = pd.read_csv('/content/sample_data/RegresiUTSTelkom.csv')

    # Menampilkan lima data pertama
    df.head()
```

	2001	49.94357	21.47114	73.0775	8.74861	-17.40628	-13.09905	-25.01202	-12.23257	7.83089	...	13.0162	-54.40548	58.99367	15.3734
0	2001	48.73215	18.42930	70.32679	12.94636	-10.32437	-24.83777	8.76630	-0.92019	18.76548	...	5.66812	-19.68073	33.04964	42.8783
1	2001	50.95714	31.85602	55.81851	13.41693	-6.57898	-18.54940	-3.27872	-2.35035	16.07017	...	3.03800	26.05866	-50.92779	10.9379
2	2001	48.24750	-1.89837	36.29772	2.58776	0.97170	-26.21683	5.05097	-10.34124	3.55005	...	34.57337	-171.70734	-16.96705	-46.6761
3	2001	50.97020	42.20998	67.09964	8.46791	-15.85279	-16.81409	-12.48207	-9.37636	12.63699	...	9.92661	-55.95724	64.92712	-17.7252
4	2001	50.54767	0.31568	92.35066	22.38696	-25.51870	-19.04928	20.67345	-5.19943	3.63566	...	6.59753	-50.69577	26.02574	18.9443

5 rows × 91 columns

Membaca dataset dan menampilkan lima data pertama.

```
# Cek nama kolom dan data
print(df.columns)
print(df.head())
```

```
Index(['2001', '49.94357', '21.47114', '73.0775', '8.74861', '-17.40628',
       '-13.09905', '-25.01202', '-12.23257', '7.83089', '-2.46783', '3.32136',
       '-2.31521', '10.20556', '611.10913', '951.0896', '698.11428',
       '408.98485', '383.70912', '326.51512', '238.11327', '251.42414',
       '187.17351', '100.42652', '179.19498', '-8.41558', '-317.87038',
       '95.86266', '48.10259', '-95.66303', '-18.06215', '1.96984', '34.42438',
       '11.7267', '1.3679', '7.79444', '-0.36994', '-133.67852', '-83.26165',
       '-37.29765', '73.04667', '-37.36684', '-3.13853', '-24.21531',
       '-13.23066', '15.93809', '-18.60478', '82.15479', '240.5798',
       '-10.29407', '31.58431', '-25.38187', '-3.90772', '13.29258', '41.5506',
       '-7.26272', '-21.00863', '105.50848', '64.29856', '26.08481',
       '-44.5911', '-8.30657', '7.93706', '-10.7366', '-95.44766', '-82.03307',
       '-35.59194', '4.69525', '70.95626', '28.09139', '6.02015', '-37.13767',
       '-41.1245', '-8.40816', '7.19877', '-8.60176', '-5.90857', '-12.32437',
       '14.68734', '-54.32125', '40.14786', '13.0162', '-54.40548', '58.99367',
       '15.37344', '1.11144', '-23.08793', '68.40795', '-1.82223', '-27.46348',
       '2.26327'],
      dtype='object')
   2001  49.94357  21.47114   73.0775   8.74861  -17.40628  -13.09905  \
0  2001  48.73215  18.42930  70.32679  12.94636  -10.32437  -24.83777
1  2001  50.95714  31.85602  55.81851  13.41693   -6.57898  -18.54940
2  2001  48.24750  -1.89837  36.29772   2.58776    0.97170  -26.21683
3  2001  50.97020  42.20998  67.09964   8.46791  -15.85279  -16.81409
4  2001  50.54767   0.31568  92.35066  22.38696  -25.51870  -19.04928

   -25.01202  -12.23257   7.83089  ...   13.0162  -54.40548  58.99367  \
0    8.76630   -0.92019  18.76548  ...   5.66812  -19.68073  33.04964
1   -3.27872   -2.35035  16.07017  ...   3.03800   26.05866 -50.92779
2    5.05097  -10.34124   3.55005  ...  34.57337 -171.70734 -16.96705
```

Mengecek nama kolom dan data

psi Data

```
[ ]  df.describe()
```

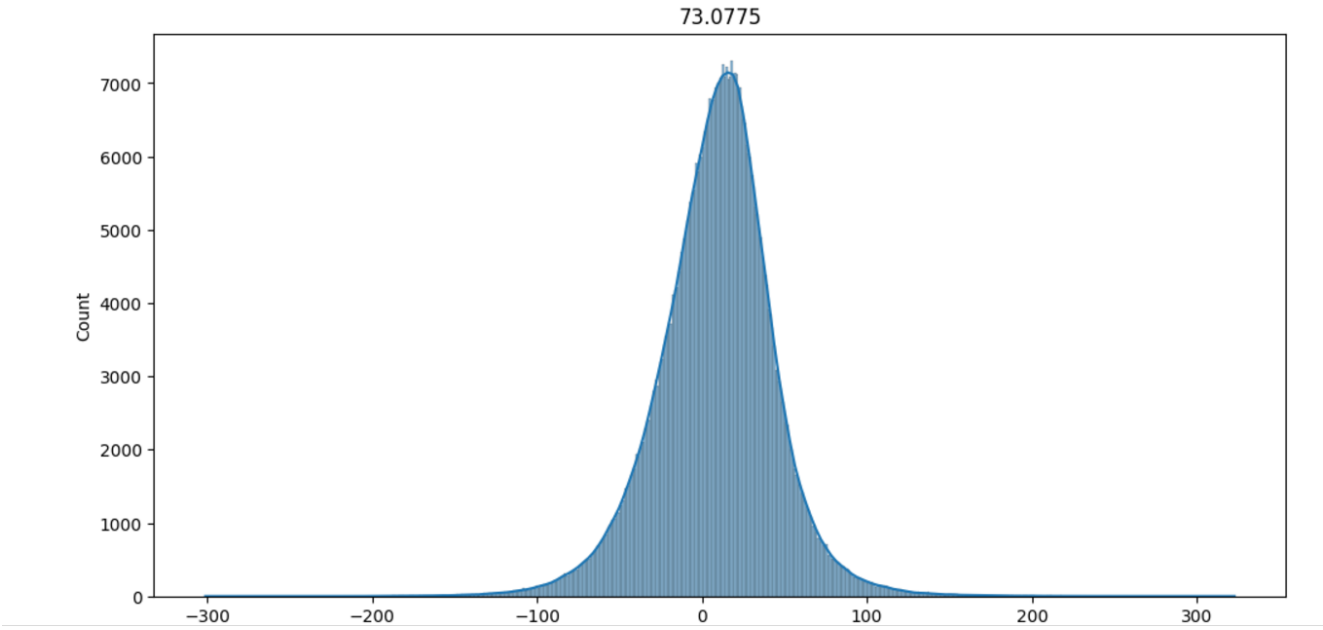
	2001	49.94357	21.47114	73.0775	8.74861	-17.40628	-13.09905	-25.01202	-12.23257	
count	515344.000000	515344.000000	515344.000000	515344.000000	515344.000000	515344.000000	515344.000000	515344.000000	515344.000000	51534
mean	1998.397077	43.387113	1.289515	8.658222	1.164110	-6.553580	-9.521968	-2.391046	-1.793215	
std	10.931056	6.067557	51.580393	35.268505	16.322802	22.860803	12.857763	14.571853	7.963822	1
min	1922.000000	1.749000	-337.092500	-301.005060	-154.183580	-181.953370	-81.794290	-188.214000	-72.503850	-12
25%	1994.000000	39.954667	-26.059848	-11.462775	-8.487507	-20.666455	-18.441005	-10.780360	-6.468390	-
50%	2002.000000	44.258490	8.417725	10.476235	-0.652855	-6.007770	-11.188355	-2.046625	-1.736415	
75%	2006.000000	47.833875	36.124030	29.764685	8.787548	7.741877	-2.388945	6.508587	2.913455	
max	2011.000000	61.970140	384.065730	322.851430	335.771820	262.068870	166.236890	172.402680	126.741270	14

8 rows × 91 columns

Memberikan ringkasan statistik deskriptif untuk setiap kolom numerik dalam dataset, seperti nilai rata□rata (mean), standar deviasi (std), nilai minimum (min), kuartil (25%, 50%, 75%), dan nilai

```
plt.figure(figsize=(12,6))
sns.histplot(df['73.0775'], kde=True)
plt.title("73.0775")
```

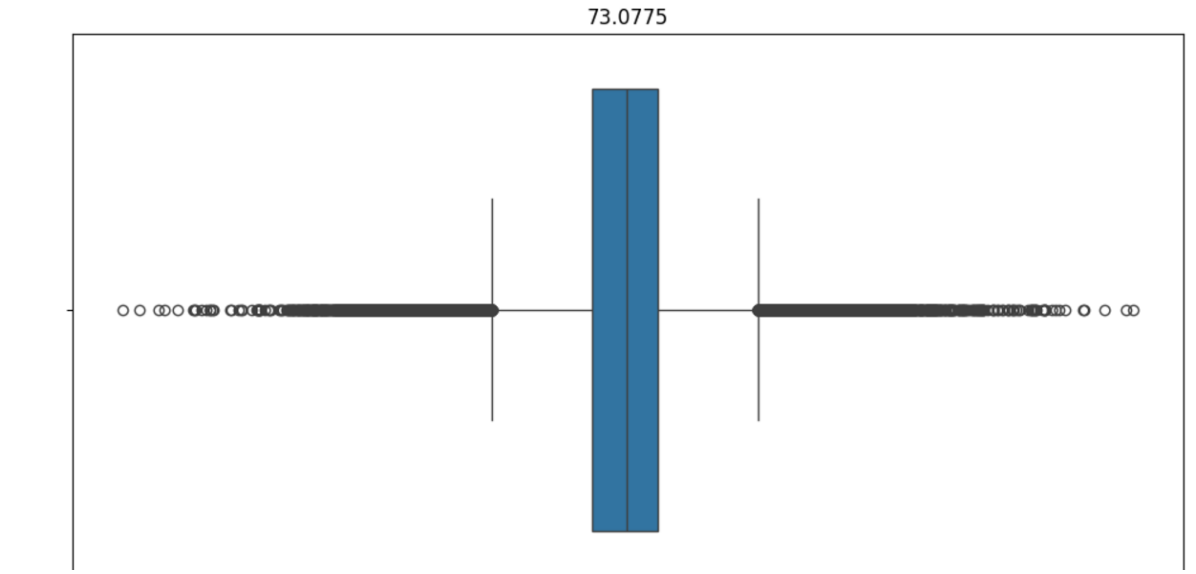
```
Text(0.5, 1.0, '73.0775')
```



Menampilkan distribusi data kolom '73.0775' menggunakan histogram dengan kurva KDE (Kernel Density Estimation).

```
# Boxplot untuk melihat outlier
plt.figure(figsize=(12,6))
sns.boxplot(x=df['73.0775'])
plt.title("73.0775")
```

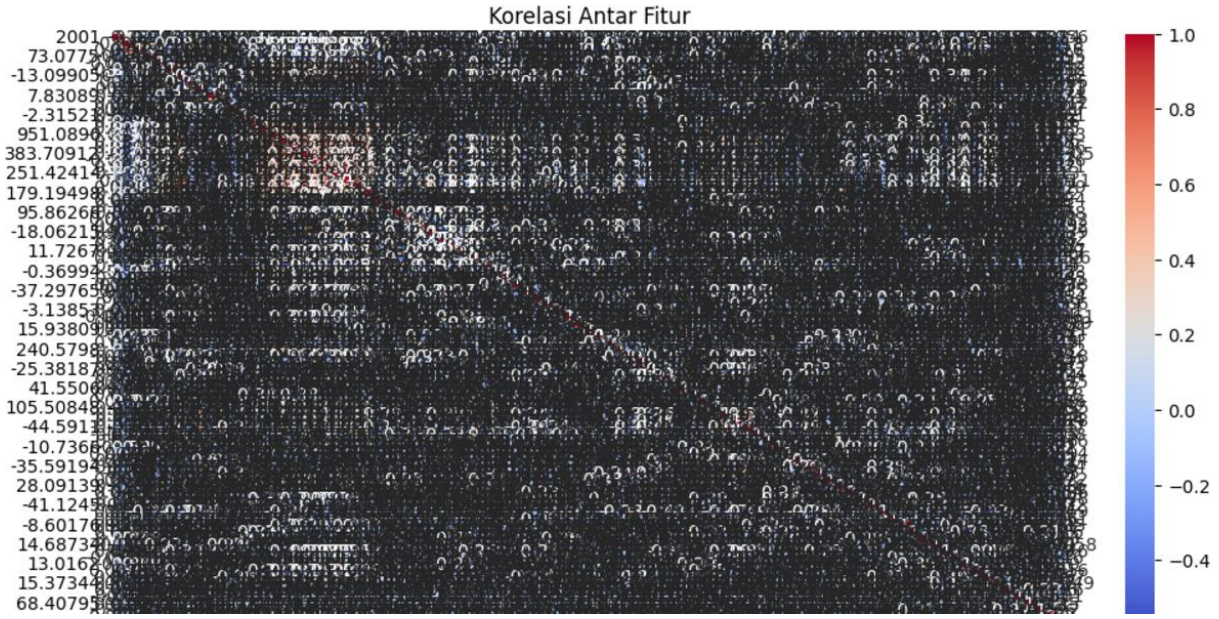
```
Text(0.5, 1.0, '73.0775')
```



Mengidentifikasi outlier dalam kolom '73.0775' menggunakan boxplot.

```
[ ] plt.figure(figsize=(12,6))
    sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
    plt.title("Korelasi Antar Fitur")
```

```
Text(0.5, 1.0, 'Korelasi Antar Fitur')
```



Menampilkan korelasi antar fitur numerik dalam dataset.

Data Preprocessing dan Split Data

```
[ ] from sklearn.preprocessing import PolynomialFeatures
    from sklearn.impute import SimpleImputer
    from sklearn.model_selection import train_test_split

    # Contoh data (sesuaikan dengan dataset)
    import pandas as pd
    import numpy as np
    data = pd.DataFrame({
        'feature1': ['2001', '49.94357', '21.47114', '40.14786', '13.0162'],
        'feature2': ['41.5506', '1.11144', '1.3679', '11.7267', '1.3679'],
    })
```

```
[ ] # Pisahkan fitur (X) dan target (y) jika perlu
    X = data[['feature1', 'feature2']]
    y = [1, 0, 1, 0, 1]
```

```
[ ] # Menangani nilai yang hilang
    from sklearn.impute import SimpleImputer
    imputer = SimpleImputer(strategy='mean')
    X_imputed = imputer.fit_transform(X)
```

```
[ ] # Pastikan dimensi sesuai
    print(X_imputed.shape)  # Harus (5, 2) karena ada 5 baris dan 2 fitur
    print(len(y))  # Harus 5 elemen
```

```
(5, 2)
5
```

- Membuat contoh dataset dengan dua fitur, `feature1` dan `feature2`. Data dalam bentuk string harus dikonversi ke tipe numerik untuk analisis.
- X: Memisahkan fitur independen (predictor), y: Membuat label target untuk klasifikasi.
- Mengisi nilai kosong (NaN) dalam fitur dengan rata-rata nilai kolom.
- Memastikan dimensi data fitur (X_imputed) dan target (y) sesuai.

```
[ ] # Pisahkan data latih dan data uji
    from sklearn.model_selection import train_test_split
    X_train_imputed, X_test_imputed, y_train, y_test = train_test_split(X_imputed, y, test_size=0.2, random_state=42)
```

```
[ ] # Memeriksa apakah masih ada nilai NaN setelah imputasi
    print(pd.DataFrame(X_train_imputed).isnull().sum())
    print(pd.DataFrame(X_test_imputed).isnull().sum())
```

```
0    0
1    0
dtype: int64
0    0
1    0
dtype: int64
```

```
[ ] print(X_train_imputed.shape, X_test_imputed.shape)
    print(len(y_train), len(y_test))
```

```
(4, 2) (1, 2)
4 1
```

- Membagi data menjadi 80% untuk pelatihan dan 20% untuk pengujian.
- Memastikan tidak ada nilai NaN setelah proses imputasi.
- Memeriksa dimensi set pelatihan dan pengujian untuk memastikan pembagian dilakukan dengan benar.

Pipeline: Polynomial Regression

```
[ ] from sklearn.preprocessing import PolynomialFeatures

    # Polynomial Regression
    poly = PolynomialFeatures(degree=3)
    X_poly_train = poly.fit_transform(X_train_imputed)
    X_poly_test = poly.transform(X_test_imputed)
```

```
[ ] from sklearn.linear_model import LinearRegression

    # Polynomial Regression
    poly_model = LinearRegression()
    poly_model.fit(X_poly_train, y_train)
```

```
LinearRegression
LinearRegression()
```

```python
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
import numpy as np

# Prediksi dan Evaluasi
y_poly_pred = poly_model.predict(X_poly_test)
print("Polynomial Regression - R2:", r2_score(y_test, y_poly_pred))
print("Polynomial Regression - MAE:", mean_absolute_error(y_test, y_poly_pred))
print("Polynomial Regression - RMSE:", np.sqrt(mean_squared_error(y_test, y_poly_pred)))
```

```
Polynomial Regression - R2: nan
Polynomial Regression - MAE: 1.1003006969996485
Polynomial Regression - RMSE: 1.1003006969996485
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less t
  warnings.warn(msg, UndefinedMetricWarning)
```