**Mudah Belajar Otodidak Data Science**

**(Praktek Menggunakan Python3)**

**Edisi 2 Tahun 2023**

**Disusun Oleh:**

**Aryajaya Alasmsyah, S.Kom., M.Kom., MTA.**

**Materi Praktikum Data Science**

1. Pengantar Bahasa Python
2. Visualisasi Data Menggunakan Matplotlib dan Seaborn
3. Visualisasi Data Menggunakan Framework Streamlit
4. Teknik-Teknik Praproses Data – Data Tabular, Time Series, dan Spasial
5. Algoritma Klasifikasi Data Bagian 1
6. Algoritma Klastering Data Bagian 2
7. Ujian Tengah Semester (UTS)
8. Algoritma Klastering Data Bagian 1
9. Algoritma Klastering Data Bagian 2
10. Algoritma Regresi Linier
11. Algoritma Neural Network: SBi-LSTM dan SBi-GRU
12. Algoritma Neural Network: SBi-LSTM-XGBoost dan SBi-GRU-XGBoostost
13. Presentasi Projek
14. Ujian Akhir Semester (UAS)

|  |
| --- |
| **Pertemuan 10 – Regresi Linier dengan OLS** |

|  |
| --- |
| **Tujuan pembelajaran**   * Mahasiswa mampu memahami konsep dasar regresi linier sederhana. * Mahasiswa mampu memahami konsep dasar regresi linier berganda. * Mahasiswa mampu menggunakan library sklearn dan statsmodels. |

**Studi Kasus: Regression Liniear with Dummy Dataset**

|  |
| --- |
| # lib manipulation dataset  import numpy as np  import pandas as pd    # lib visualization data  import matplotlib.pyplot as plt  import seaborn as sns    # lib regression linear  import statsmodels.api as sm  import statsmodels.formula.api as smf  from scipy.stats import pearsonr  from sklearn.linear\_model import LinearRegression |

1. **Akuisisi Data**

|  |
| --- |
| # load dataset  dataset = pd.read\_csv("../dataset/dummy\_regresi.csv", parse\_dates=["tahun"])  dataset = dataset.set\_index("tahun")  dataset.info() |
| <class 'pandas.core.frame.DataFrame'>  DatetimeIndex: 24 entries, 2016-01-01 to 2017-12-01  Data columns (total 3 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----  0 x1 24 non-null float64  1 x2 24 non-null float64  2 y 24 non-null int64  dtypes: float64(2), int64(1)  memory usage: 768.0 bytes |
| # show metadata  print(np.round(dataset.describe(),2)) |
| x1 x2 y  count 24.00 24.00 24.00  mean 2.07 5.78 1070.08  std 0.35 0.33 210.74  min 1.75 5.30 704.00  25% 1.75 5.50 928.25  50% 2.00 5.85 1061.00  75% 2.50 6.10 1239.00  max 2.75 6.20 1464.00 |

1. **Explorasi Data Analisis**

|  |
| --- |
| # func lineplot for timeseries  def lineplot(x, y, label, title):      # time series plot    fig, ax = plt.subplots(figsize = (8,4))    ax.plot(x, y, color="tab:blue", label=label, linewidth=2, marker='o')      # set label-labels    ax.set\_title(title,fontsize=14)    ax.set\_xlabel("",fontsize=12)    ax.set\_ylabel("",fontsize=12)    ax.legend(loc="best")    ax.grid(True)      # rotasi label x menjadi 45 derajat    plt.setp(ax.xaxis.get\_majorticklabels(), rotation=30)    plt.setp(ax.xaxis.get\_majorticklabels(), rotation=30)    plt.tight\_layout()      # return values    return plt.show() |

|  |
| --- |
| # resample 4 monthly  df\_resample = dataset.resample('4ME', closed='left').sum()  print(df\_resample) |
| x1 x2 y  tahun  2016-04-30 7.00 24.4 3121  2016-08-31 7.00 24.5 3670  2016-12-31 7.25 23.8 3913  2017-04-30 8.50 22.7 4531  2017-08-31 9.75 22.0 4939  2017-12-31 10.25 21.2 5508 |
| # visualization x1  lineplot(  x=df\_resample.index, y=df\_resample["x1"],  label="variable x1", title="Visualization of Time Series X1") |
|  |
| # visualization x2  lineplot(  x=df\_resample.index, y=df\_resample["x2"],  label="variable x2", title="Visualization of Time Series X2") |
|  |
| # visualization y  lineplot(  x=df\_resample.index, y=df\_resample["y"],  label="variable y", title="Visualization of Time Series Y") |
|  |
| Gambar x. Output program |

|  |
| --- |
| # function scatter plot  def scatter(x, y, label, title):      # create scatter plots    fig, ax = plt.subplots(figsize = (8,4))    ax.scatter(x, y, color='tab:blue', label=label, linewidth=2)      # set labels    ax.set\_title(title, fontsize=14)    ax.set\_xlabel("", fontsize=12)    ax.set\_ylabel("", fontsize=12)    ax.legend(loc='upper left')    ax.grid(True)      # return values    plt.tight\_layout()    plt.show() |

|  |
| --- |
| # Scatterplot  scatter(  x=dataset["x1"], y=dataset["y"],  label="Linearity of X2 with Y", title="Scatterplot X1 with Y") |
|  |
| # Scatterplot  scatter(  x=dataset["x2"], y=dataset["y"],  label="Linearity of X2 with Y", title="Scatterplot X2 with Y") |
|  |
| Gambar x. Output program |

1. **Linear Regression menggunakan Scikit Learn dan OLS**

|  |
| --- |
| # function regression  def results\_reg(xtrue, ytrue, ypred, label1, label2, title):      # plot results regression    fig, ax = plt.subplots(figsize = (8,4))    ax.scatter(xtrue, ytrue, color='tab:blue', label=label1, linewidth=2)    ax.plot(xtrue, ypred, color="tab:orange", label=label2, linewidth=2)      # set labels    ax.set\_title(title, fontsize=14)    ax.set\_xlabel("", fontsize=12)    ax.set\_ylabel("", fontsize=12)    ax.legend(loc='upper left')    ax.grid(True)      # return values    plt.tight\_layout()    plt.show() |

|  |
| --- |
| # Bivariate. X1 with Y  model1 = LinearRegression().fit(  np.array(dataset["x1"]).reshape(-1,1), np.array(dataset["y"]).reshape(-1,1)  )    # predict with x1  yhat1 = model1.predict(np.array(dataset["x1"]).reshape(-1,1))    # Bivariate. X1 with Y  model2 = LinearRegression().fit(  np.array(dataset["x2"]).reshape(-1,1), np.array(dataset["y"]).reshape(-1,1)  )    # predict with x2  yhat2 = model2.predict(np.array(dataset["x2"]).reshape(-1,1)) |
|  |
| # results regression x1 with y  results\_reg(    xtrue=dataset["x1"], ytrue=dataset["y"], ypred=yhat1,    label1="Actual data", label2="Prediction",    title="Results Regression Liniear X1 with Y"  ) |
|  |
|  |
| Gambar x. Output program |
|  |
| # results regression x2 with y  results\_reg(    xtrue=dataset["x2"], ytrue=dataset["y"], ypred=yhat2,    label1="Actual data", label2="Prediction",    title="Results Regression Liniear X2 with Y"  ) |
|  |
|  |
| Gambar x. Output program |

1. **Linear Regression via Statsmodel + OLS**

|  |
| --- |
| # linier regression  model\_lin = smf.ols(formula="y ~ x1", data=dataset).fit()  print(model\_lin.summary()) |
| OLS Regression Results  ==============================================================================  Dep. Variable: y R-squared: 0.876  Model: OLS Adj. R-squared: 0.870  Method: Least Squares F-statistic: 155.0  Date: Tue, 02 Jul 2024 Prob (F-statistic): 1.95e-11  Time: 02:55:07 Log-Likelihood: -136.94  No. Observations: 24 AIC: 277.9  Df Residuals: 22 BIC: 280.2  Df Model: 1  Covariance Type: nonrobust  ==============================================================================  coef std err t P>|t| [0.025 0.975]  ------------------------------------------------------------------------------  Intercept -99.4643 95.210 -1.045 0.308 -296.918 97.990  x1 564.2039 45.317 12.450 0.000 470.221 658.186  ==============================================================================  Omnibus: 5.361 Durbin-Watson: 0.514  Prob(Omnibus): 0.069 Jarque-Bera (JB): 3.647  Skew: -0.927 Prob(JB): 0.161  Kurtosis: 3.459 Cond. No. 15.8  ============================================================================== |

|  |
| --- |
| # linier regression  model\_lin = smf.ols(formula="y ~ x2", data=dataset).fit()  print(model\_lin.summary()) |
| OLS Regression Results  ==============================================================================  Dep. Variable: y R-squared: 0.851  Model: OLS Adj. R-squared: 0.844  Method: Least Squares F-statistic: 125.4  Date: Tue, 02 Jul 2024 Prob (F-statistic): 1.49e-10  Time: 02:55:07 Log-Likelihood: -139.14  No. Observations: 24 AIC: 282.3  Df Residuals: 22 BIC: 284.6  Df Model: 1  Covariance Type: nonrobust  ==============================================================================  coef std err t P>|t| [0.025 0.975]  ------------------------------------------------------------------------------  Intercept 4471.3393 304.254 14.696 0.000 3840.354 5102.324  x2 -588.9621 52.602 -11.196 0.000 -698.053 -479.871  ==============================================================================  Omnibus: 0.397 Durbin-Watson: 1.083  Prob(Omnibus): 0.820 Jarque-Bera (JB): 0.535  Skew: -0.105 Prob(JB): 0.765  Kurtosis: 2.299 Cond. No. 107.  ============================================================================== |

|  |
| --- |
| # linier regression  model\_lin = smf.ols(formula="y ~ x1 + x2", data=dataset).fit()  print(model\_lin.summary()) |
| OLS Regression Results  ==============================================================================  Dep. Variable: y R-squared: 0.898  Model: OLS Adj. R-squared: 0.888  Method: Least Squares F-statistic: 92.07  Date: Tue, 02 Jul 2024 Prob (F-statistic): 4.04e-11  Time: 02:55:07 Log-Likelihood: -134.61  No. Observations: 24 AIC: 275.2  Df Residuals: 21 BIC: 278.8  Df Model: 2  Covariance Type: nonrobust  ==============================================================================  coef std err t P>|t| [0.025 0.975]  ------------------------------------------------------------------------------  Intercept 1798.4040 899.248 2.000 0.059 -71.685 3668.493  x1 345.5401 111.367 3.103 0.005 113.940 577.140  x2 -250.1466 117.950 -2.121 0.046 -495.437 -4.856  ==============================================================================  Omnibus: 2.691 Durbin-Watson: 0.530  Prob(Omnibus): 0.260 Jarque-Bera (JB): 1.551  Skew: -0.612 Prob(JB): 0.461  Kurtosis: 3.226 Cond. No. 394.  ============================================================================== |

**Selesai, Selamat Mencoba :3**