|  |
| --- |
| **Pertemuan 4 – Teknik Praproses Data** |
|  |
| **Tujuan pembelajaran** |
| * Mahasiswa mampu menggunakan library scikit-learn untuk praproses data. * Mahasiswa mampu menerapkan beberapa teknik praproses data seperti pembersihan data, normalisasi data, penggabungan data, reduksi data, dan pembagian data. |

Data Tabular – Visualisasi Iris Datataset

|  |
| --- |
| # lib manipulasi data  import pandas as pd  import numpy as np    # lib visualisasi data  import seaborn as sns  import matplotlib.pyplot as plt    # lib preprocessing  from sklearn.preprocessing import MinMaxScaler  from sklearn.preprocessing import StandardScaler  from sklearn.model\_selection import train\_test\_split |
|  |
| # load dataset  dataset = pd.read\_csv("../dataset/dataset\_iris.csv") |
|  |
| # Set features and Labels  x = dataset[["sepal\_length","sepal\_width","petal\_length","petal\_width"]].values  y = dataset["species"].values |

**MinMax - Scaler**

|  |
| --- |
| # process normalization data  scaler = MinMaxScaler(feature\_range=(0,1))  scaled = scaler.fit\_transform(x) |
|  |
| # results normalization data  minmax = pd.concat([    pd.DataFrame(scaled, columns=["sepal\_length","sepal\_width","petal\_length","petal\_width"]),    pd.DataFrame(y, columns=["species"])  ],axis=1) |
| # results normalization data  print(minmax.tail()) |
|  |
| sepal\_length sepal\_width petal\_length petal\_width species  145 0.666667 0.416667 0.711864 0.916667 virginica  146 0.555556 0.208333 0.677966 0.750000 virginica  147 0.611111 0.416667 0.711864 0.791667 virginica  148 0.527778 0.583333 0.745763 0.916667 virginica  149 0.444444 0.416667 0.694915 0.708333 virginica |

**Standard - Scaler**

|  |
| --- |
| # process normalization data  scaled = StandardScaler().fit\_transform(x) |
|  |
| # results normalization data  standard = pd.concat([    pd.DataFrame(scaled, columns=["sepal\_length","sepal\_width","petal\_length","petal\_width"]),    pd.DataFrame(y, columns=["species"])  ],axis=1) |
|  |
| # results normalization data  print(standard.tail()) |
|  |
| sepal\_length sepal\_width petal\_length petal\_width species  145 1.038005 -0.124958 0.819624 1.447956 virginica  146 0.553333 -1.281972 0.705893 0.922064 virginica  147 0.795669 -0.124958 0.819624 1.053537 virginica  148 0.432165 0.800654 0.933356 1.447956 virginica  149 0.068662 -0.124958 0.762759 0.790591 virginica |

|  |  |
| --- | --- |
| # function boxplot  def boxplot(dataset, title):      # create boxplot with seaborn    fig, ax = plt.subplots(figsize=(8,4))    sns.boxplot(data=dataset, x="species", y="petal\_length", hue="species")      # set labels    ax.set\_title(title, fontsize=12)    ax.set\_xlabel("", fontsize=12)    ax.set\_ylabel("", fontsize=12)    ax.grid(True)      # return values    plt.tight\_layout()    plt.show() | |
|  | |
| # call function boxplot  boxplot(minmax, "Boxplot with min-max scaler")  boxplot(standard, "Boxplot with standard scaler") | |
|  | |
|  |  |
| Gambar x. Output program | |

|  |  |
| --- | --- |
| # function heatmap  def heatmap(dataset, title):      # create heatmap with seaborn    fig, ax = plt.subplots(figsize=(8,4))    sns.heatmap(      data=dataset.corr(numeric\_only=True), vmin=-1, vmax=1, cmap="viridis", annot=True)      # set labels    ax.set\_title(title, fontsize=12)    ax.grid(False)      # show plot    plt.tight\_layout()    plt.show() | |
|  | |
| # call function heatmap  heatmap(minmax, "Heatmap with min-max scaler")  heatmap(standard, "Heatmap with standard scaler") | |
|  | |
|  |  |
| Gambar x. Output program | |

|  |  |
| --- | --- |
| # function scatter plot  def scatter(data, x, y, title):      # create scatter plots    fig, ax = plt.subplots(figsize = (8,4))    sns.scatterplot(data=data, x=x, y=y, hue="species")      # set labels    ax.set\_title(title, fontsize=12)    ax.set\_xlabel("", fontsize=12)    ax.set\_ylabel("", fontsize=12)    ax.grid(True)      # return values    plt.tight\_layout()    plt.show() | |
|  | |
| # call func scatterplot  scatter(minmax, "petal\_length", "petal\_width", "Heatmap with min-max scaler")  scatter(standard, "petal\_length", "petal\_width", "Heatmap with standard scaler") | |
|  | |
|  |  |
| Gambar x. Output program | |

Data Time Series– Visualisasi Cryptocurrency dan Stock Price

|  |
| --- |
| # lib manipulasi data  import pandas as pd  import numpy as np    # lib untuk visualisasi data  import seaborn as sns  import matplotlib.pyplot as plt    # lib praproses data  from sklearn.preprocessing import MinMaxScaler  from sklearn.model\_selection import train\_test\_split |

|  |
| --- |
| # load dataset  dataset = pd.read\_csv("../dataset/Cryptocurrency-BTC-USD-2024-05.csv", parse\_dates=['Date'])  dataset = dataset.set\_index("Date") |
|  |
| # show metadataset  print(np.round(    dataset[["Open","High","Low","Close"]].describe(),0  )) |
|  |
| Open High Low Close  count 3439.0 3439.0 3439.0 3439.0  mean 16939.0 17333.0 16520.0 16958.0  std 18329.0 18761.0 17855.0 18345.0  min 177.0 212.0 172.0 178.0  25% 1724.0 1787.0 1656.0 1729.0  50% 9146.0 9300.0 8956.0 9159.0  75% 27807.0 28361.0 27268.0 27872.0  max 73079.0 73750.0 71334.0 73084.0 |
|  |
| # round .3f  dataset = np.round(dataset[["Open","High","Low","Close"]],4)    # show dataset  print(dataset) |
|  |
| Open High Low Close  Date  2015-01-01 320.4350 320.4350 314.0030 314.2490  2015-01-02 314.0790 315.8390 313.5650 315.0320  2015-01-03 314.8460 315.1500 281.0820 281.0820  2015-01-04 281.1460 287.2300 257.6120 264.1950  2015-01-05 265.0840 278.3410 265.0840 274.4740  ... ... ... ... ...  2024-05-27 68512.1797 70597.8828 68232.5000 69394.5547  2024-05-28 69392.1953 69514.6406 67227.1562 68296.2188  2024-05-29 68296.3516 68852.4609 67101.4922 67578.0938  2024-05-30 67576.0859 69500.5391 67118.0781 68364.9922  2024-05-31 68362.5156 68999.5625 66633.4219 67491.4141  [3439 rows x 4 columns] |

|  |
| --- |
| # time series plot  fig, ax = plt.subplots(figsize = (8,4))  ax.plot(dataset.index.values, dataset["Open"], color="tab:green", label="Open", linewidth=2)  ax.plot(dataset.index.values, dataset["High"], color="tab:orange", label="High", linewidth=2)  ax.plot(dataset.index.values, dataset["Low"], color="tab:red", label="Low", linewidth=2)  ax.plot(dataset.index.values, dataset["Close"], color="tab:blue", label="Close", linewidth=2)    # set label-labels  ax.set\_title("", fontsize=14)  ax.set\_xlabel("", fontsize=12)  ax.set\_ylabel("", fontsize=12)  ax.legend(loc="best")  ax.grid(True)  plt.show() |
|  |
|  |
| Gambar x. Output program |

|  |
| --- |
| # convert dataframe to series close price  data = dataset.values  np.round(data[:3],7) |
|  |
| array([[320.435, 320.435, 314.003, 314.249],  [314.079, 315.839, 313.565, 315.032],  [314.846, 315.15 , 281.082, 281.082]]) |
|  |
| # normalize features  scaler = MinMaxScaler(feature\_range=(0, 1))  scaled\_data = scaler.fit\_transform(np.array(data)) |
|  |
| # show normalize data  scaled\_data[:3] |
|  |
| array([[0.0019689 , 0.00147819, 0.00200236, 0.00186743],  [0.00188172, 0.0014157 , 0.0019962 , 0.00187817],  [0.00189224, 0.00140633, 0.00153974, 0.0014125 ]]) |
|  |
| # time series plot  fig, ax = plt.subplots(figsize = (8,4))  ax.plot(dataset.index, scaled\_data[:,0:1], color="tab:green", label="Open", linewidth=2)  ax.plot(dataset.index, scaled\_data[:,1:2], color="tab:orange", label="High", linewidth=2)  ax.plot(dataset.index, scaled\_data[:,2:3], color="tab:red", label="Low", linewidth=2)  ax.plot(dataset.index, scaled\_data[:,3:4], color="tab:blue", label="Close", linewidth=2)    # set label-labels  ax.set\_title("",fontsize=14)  ax.set\_xlabel("",fontsize=12)  ax.set\_ylabel("",fontsize=12)  ax.legend(loc="best")  ax.grid(True)  plt.show() |
|  |
|  |
| Gambar x. Output program |

|  |
| --- |
| # results preprocessing of normalize data  df\_normalized = pd.concat([    pd.DataFrame(dataset.index.values, columns=["Date"]),    pd.DataFrame(scaled\_data, columns=["Open", "High", "Low", "Close"]),  ], axis=1) |
|  |
| # split data train and test  train\_data, test\_data = train\_test\_split(df\_normalized, train\_size=0.80, test\_size=0.20,  shuffle=False) |
|  |
| # create time series plot  fig, ax = plt.subplots(figsize = (8,4))  ax.plot(    df\_normalized["Date"].iloc[0:len(train\_data)], train\_data["Close"], color="tab:blue",  label="training", linewidth=2)  ax.plot(    df\_normalized["Date"].iloc[len(train\_data):], test\_data["Close"], color="tab:red",  label="testing", linewidth=2)    # set labels  ax.set\_title("",fontsize=14)  ax.set\_xlabel("",fontsize=12)  ax.set\_ylabel("",fontsize=12)  ax.legend(loc="best")  ax.grid(True)    # show plot  plt.show() |
|  |
|  |
| Gambar x. Output program |

**Selesai, Selamat Mencoba :3**