|  |
| --- |
| **Pertemuan 2 – Visualisasi Seaborn dan Matplotlib** |
|  |
| **Tujuan pembelajaran** |
| * Mahasiswa mampu menggunakan library seperti seaborn dan matplotlib. * Mahasiswa mampu membuat visualisasi data seperti barplot, lineplot, scatterplot, histogram, pieplot, boxplot, heatmap, dan lain-lain. |

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 |
|  |
| # load dataset  dataset = pd.read\_csv("../../dataset/dataset\_iris.csv")  print(dataset.info()) |
|  |
| <class 'pandas.core.frame.DataFrame'>  RangeIndex: 150 entries, 0 to 149  Data columns (total 5 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----  0 sepal\_length 150 non-null float64  1 sepal\_width 150 non-null float64  2 petal\_length 150 non-null float64  3 petal\_width 150 non-null float64  4 species 150 non-null object |

|  |
| --- |
| # create barplot  fig, ax = plt.subplots(figsize=(8,4))  sns.countplot(dataset, x="species", hue="species")    # show plot  plt.grid(True)  plt.tight\_layout()  plt.show() |
|  |
|  |
| Gambar x. Output program |

|  |
| --- |
| # create pie chart  fig, ax = plt.subplots(figsize=(8,4))    # set name of class  class\_names = ['setosa', 'versicolor', 'virginica']    # calculate number labels  class\_counts = dataset["species"].value\_counts()    # calculate percentage  percentages = (class\_counts / class\_counts.sum()) \* 100    # pieplot with percentage labels  ax.pie(x=percentages, labels=class\_names, explode=(0.01, 0.01, 0.01), autopct="%.2f%%")    # show plot  plt.axis('scaled')  plt.tight\_layout()  plt.show() |
|  |
|  |
| Gambar x. Output program |

|  |
| --- |
| # define boxplot  fig, ax = plt.subplots(figsize=(8,4))  ax.boxplot(    dataset[["sepal\_length", "sepal\_width", "petal\_length", "petal\_width"]],    labels=["sepal\_length", "sepal\_width", "petal\_length", "petal\_width"],    patch\_artist=True, widths=(0.75, 0.75, 0.75, 0.75)  )    # show boxplot  plt.grid(True)  plt.tight\_layout()  plt.show() |
|  |
|  |
| Gambar x. Output program |

|  |  |
| --- | --- |
| # function scatter plot  def scatter(data, x, y):      # create scatter plots    fig, ax = plt.subplots(figsize = (8,4))    sns.scatterplot(data=data, x=x, y=y, hue="species")      # return values    plt.grid(True)    plt.tight\_layout()    plt.show() | |
|  | |
| # call func scatterplot  scatter(dataset, "sepal\_length", "sepal\_width")  scatter(dataset, "petal\_length", "petal\_width") | |
|  | |
|  |  |
| Gambar x. Output program | |

|  |
| --- |
| # create heatmap corr  fig, ax = plt.subplots(figsize=(8,4))  sns.heatmap(    dataset.corr(numeric\_only=True), vmin=-1, vmax=1,    cmap="viridis", annot=True, fmt=".3f", linewidths=1  )    # show plot  plt.tight\_layout()  plt.show() |
|  |
|  |
| Gambar x. Output program |

|  |
| --- |
| # create figure  fig, ax = plt.subplots(nrows=2, ncols=2, figsize=(10,8))    # create kdeplot  sns.kdeplot(data=dataset, x="sepal\_length", hue="species", fill=True, ax=ax[0][0])  sns.kdeplot(data=dataset, x="sepal\_width", hue="species", fill=True, ax=ax[0][1])  sns.kdeplot(data=dataset, x="petal\_length", hue="species", fill=True, ax=ax[1][0])  sns.kdeplot(data=dataset, x="petal\_width", hue="species", fill=True, ax=ax[1][1])    # show plots  plt.tight\_layout()  plt.show() |
|  |
|  |
| Gambar x. Output program |

|  |
| --- |
| # create figure  fig, ax = plt.subplots(nrows=2, ncols=2, figsize=(10,8))    # create kdeplot  sns.kdeplot(data=dataset, x="sepal\_length", y="sepal\_width", hue="species", ax=ax[0][0])  sns.kdeplot(data=dataset, x="petal\_length", y="sepal\_width", hue="species", ax=ax[0][1])  sns.kdeplot(data=dataset, x="sepal\_length", y="petal\_width", hue="species", ax=ax[1][0])  sns.kdeplot(data=dataset, x="petal\_length", y="sepal\_length", hue="species", ax=ax[1][1])    # show plots  plt.tight\_layout()  plt.show() |
|  |
|  |
| Gambar x. Output program |

Data Time Series– Visualisasi Cryptocurrency dan Stock Price

|  |
| --- |
| # lib manipulasi data  import pandas as pd  import numpy as np    # lib visualisasi data  import matplotlib.pyplot as plt  import matplotlib.dates as mdates |
|  |
| # func load dataset  def load\_dataset(df):      # load dataset    dataset = pd.read\_csv("../../dataset/"+df, parse\_dates=['Date'])      # set feature    dataset = dataset[["Date", "Open", "High", "Low", "Close"]]      # set index    dataset = dataset.set\_index("Date")      # return values    return dataset |
|  |
| # func visualization of time series plot  def timeseries\_plot(date, data, title):      # create frame    fig, ax = plt.subplots(figsize = (8,4))      # time series plot    ax.plot(date, data[:,0:1], color="tab:green", label="Open Price", linewidth=2)    ax.plot(date, data[:,1:2], color="tab:orange", label="High Price", linewidth=2)    ax.plot(date, data[:,2:3], color="tab:red", label="Low Price", linewidth=2)    ax.plot(date, data[:,3:4], color="tab:blue", label="Close Price", linewidth=2)      # 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)      # return values    plt.show() |

|  |
| --- |
| # load dataset  df\_btc = load\_dataset("Cryptocurrency-BTC-USD-2024-05.csv")    # set features  features = np.array(df\_btc[["Open", "High", "Low", "Close"]])    # visualization btc-usd  timeseries\_plot(    date = df\_btc.index, data = features, title = "Time Series Plot of BTC-USD Price") |
|  |
|  |
| Gambar x. Output program |

|  |
| --- |
| # load dataset  df\_eth = load\_dataset("Cryptocurrency-ETH-USD-2024-05.csv")    # set features  features = np.array(df\_eth[["Open", "High", "Low", "Close"]])    # visualization df\_eth  timeseries\_plot(    date = df\_eth.index, data = features, title = "Time Series Plot of ETH-USD Price") |
|  |
|  |
| Gambar x. Output program |

|  |
| --- |
| # load dataset  df\_aapl = load\_dataset("Stock-Price-AAPL-2024-05.csv")    # set features  features = np.array(df\_aapl[["Open", "High", "Low", "Close"]])    # visualization df\_aapl  timeseries\_plot(    date = df\_aapl.index, data = features, title = "Time Series Plot of Apple.Inc Price"  ) |
|  |
|  |
| Gambar x. Output program |

|  |
| --- |
| # load dataset  df\_ibm = load\_dataset("Stock-Price-IBM-2024-05.csv")    # set features  features = np.array(df\_ibm[["Open", "High", "Low", "Close"]])    # visualization df\_ibm  timeseries\_plot(    date = df\_ibm.index, data = features, title = "Time Series Plot of IBM Price"  ) |
|  |
|  |
| Gambar x. Output program |

**Selesai, Selamat Mencoba :3**