**Mudah Belajar Otodidak Data Science**

**(Praktek dalam bahasa python3)**

**Disusun oleh**

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

**Pertemuan 2 – Visualisasi Seaborn dan Matplotlib**

**Tujuan pembelajaran**

* Mahasiswa mampu menggunakan library seperti seaborn dan matplotlib
* Mahasiswa mampu membuat visualisasi data seperti barplot, lineplot, pieplot, histogram, scatterplot, 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    # lib praproses data  from sklearn.preprocessing import MinMaxScaler |

1. Akuisisi Data

|  |
| --- |
| # load dataset  dataset = pd.read\_csv("../dataset/iris.csv")    # lihat metadataset  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  dtypes: float64(4), object(1)  memory usage: 6.0+ KB |

1. Normalisasi Data

|  |
| --- |
| # Set features and Labels  x = dataset[["sepal\_length","sepal\_width","petal\_length","petal\_width"]].values  y = dataset["species"].values |
| # proses normalisasi data  scaler = MinMaxScaler(feature\_range=(0, 1))  scaled = scaler.fit\_transform(x) |
| # hasil normalisasi data  dataset = pd.concat([    pd.DataFrame(scaled, columns=["sepal\_length","sepal\_width","petal\_length","petal\_width"]),    pd.DataFrame(y, columns=["species"]),  ], axis=1) |
| # cek hasil normalisasi data  print(dataset) |
| sepal\_length sepal\_width petal\_length petal\_width species  0 0.222222 0.625000 0.067797 0.041667 setosa  1 0.166667 0.416667 0.067797 0.041667 setosa  2 0.111111 0.500000 0.050847 0.041667 setosa  3 0.083333 0.458333 0.084746 0.041667 setosa  4 0.194444 0.666667 0.067797 0.041667 setosa  .. ... ... ... ... ...  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  [150 rows x 5 columns] |

1. Visualisasi Data

|  |
| --- |
| # create barplot  fig, ax = plt.subplots(figsize=(8,4))  sns.countplot(dataset, x="species", hue="species")    # set labels  ax.set\_title("", fontsize=14)  ax.set\_xlabel("", fontsize=12)  ax.set\_ylabel("", fontsize=12)  ax.grid(True)    # show  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")      # set labels    ax.set\_title("", fontsize=14)    ax.set\_xlabel("", fontsize=12)    ax.set\_ylabel("", fontsize=12)    ax.legend(loc='upper left')    ax.grid(True)    plt.tight\_layout()      # show plots    plt.tight\_layout()    plt.show() |
| # call function scatter  scatter(dataset, "petal\_length", "sepal\_length") |
|  |
| Gambar x. Output program |
| # call function scatter  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)    # set labels  ax.set\_title("", fontsize=14)  ax.set\_xlabel("", fontsize=12)  ax.set\_ylabel("", fontsize=12)    # show plot  plt.show() |
|  |
| Gambar x. Output program |

|  |
| --- |
| # create kdeplot  fig, ax = plt.subplots(nrows=1, ncols=2, figsize=(12,4))  sns.kdeplot(data=dataset, x="sepal\_width", hue="species", fill=True, ax=ax[0])  ax[0].set\_title("", fontsize=14)  ax[0].set\_xlabel("", fontsize=12)  ax[0].set\_ylabel("", fontsize=12)  ax[0].grid(True)    # create kdeplot  sns.kdeplot(data=dataset, x="petal\_width", y="sepal\_width", hue="species", fill=True, ax=ax[1])  ax[1].set\_title("", fontsize=14)  ax[1].set\_xlabel("", fontsize=12)  ax[1].set\_ylabel("", fontsize=12)  ax[1].grid(True)    # show plots  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)  )    # set labels  ax.set\_title('')  ax.set\_xlabel('')  ax.set\_ylabel('')  ax.grid(True)    # show boxplot  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 seaborn as sns  import matplotlib.pyplot as plt    # lib praproses data  from sklearn.preprocessing import MinMaxScaler |

1. Akuisisi Data

|  |
| --- |
| # 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 |

|  |
| --- |
| # load dataset  df\_btc = load\_dataset("BTC-USD.csv")  print(df\_btc.info()) |
| <class 'pandas.core.frame.DataFrame'>  DatetimeIndex: 3439 entries, 2015-01-01 to 2024-05-31  Data columns (total 4 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----  0 Open 3439 non-null float64  1 High 3439 non-null float64  2 Low 3439 non-null float64  3 Close 3439 non-null float64  dtypes: float64(4)  memory usage: 134.3 KB |
| # load dataset  df\_eth = load\_dataset("ETH-USD.csv")  print(df\_eth.info()) |
| <class 'pandas.core.frame.DataFrame'>  DatetimeIndex: 2343 entries, 2018-01-01 to 2024-05-31  Data columns (total 4 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----  0 Open 2343 non-null float64  1 High 2343 non-null float64  2 Low 2343 non-null float64  3 Close 2343 non-null float64  dtypes: float64(4)  memory usage: 91.5 KB |

1. Normalisasi Data

|  |
| --- |
| # func load dataset  def normalized(df):      # normalize features    scaler = MinMaxScaler(feature\_range=(0, 1))    scaled = scaler.fit\_transform(np.array(df))      # return values    return scaled |

|  |
| --- |
| # normalized data  df\_btc\_norm = normalized(df\_btc)  np.round(df\_btc\_norm[:],7) |
| array([[0.0019689, 0.0014782, 0.0020024, 0.0018674],  [0.0018817, 0.0014157, 0.0019962, 0.0018782],  [0.0018922, 0.0014063, 0.0015397, 0.0014125]]) |
| # normalized data  df\_eth\_norm = normalized(df\_eth)  np.round(df\_eth\_norm[:],7) |
| array([[0.1420878, 0.1450551, 0.1422102, 0.1455932],  [0.1455981, 0.1725811, 0.1487562, 0.1692413],  [0.1696478, 0.1849899, 0.1694899, 0.185798 ]]) |

1. Visualisasi Data

|  |
| --- |
| # 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    return plt.show() |

|  |  |
| --- | --- |
| # BTC-USD  timeseries\_plot(    date = df\_btc.index.values,    data = df\_btc\_norm,    title = "Time Series Plot of BTC-USD Price"  ) | |
| # ETH-USD  timeseries\_plot(    date = df\_eth.index.values,    data = df\_eth\_norm,    title = "Time Series Plot of ETH-USD Price"  ) | |
|  |  |
| Gambar x. Output program | |

Data Spasial Time Series– Visualisasi Kebakaran Hutan di Indonesia

|  |
| --- |
| # lib manipulasi data  import pandas as pd  import numpy as np    # lib visualisasi data  import seaborn as sns  import matplotlib.pyplot as plt    # lib praproses data  from sklearn.preprocessing import MinMaxScaler |

1. Kebakaran Hutan

|  |
| --- |
| # load dataset  dataset = pd.read\_excel("../dataset/luas-karhutla-provinsi.xlsx", sheet\_name="hasil")    # lihat metadataset  print(dataset.info()) |
| <class 'pandas.core.frame.DataFrame'>  RangeIndex: 11 entries, 0 to 10  Data columns (total 2 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----  0 tahun 11 non-null int64  1 luas 11 non-null int64  dtypes: int64(2)  memory usage: 308.0 bytes |

|  |
| --- |
| # create frame  fig, ax = plt.subplots(figsize = (8,4))  sns.barplot(dataset, x="tahun", y="luas", color="tab:blue")    # 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)    # return values  plt.show() |
|  |
| Gambar x. Output program |

1. Titik Panas

|  |
| --- |
| # load dataset  dataset = pd.read\_csv("../dataset/fire\_archive\_M-C61\_484133.csv", parse\_dates=["acq\_date"])    # lihat metadataset  print(dataset.info()) |
| <class 'pandas.core.frame.DataFrame'>  RangeIndex: 1490972 entries, 0 to 1490971  Data columns (total 15 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----  0 latitude 1490972 non-null float64  1 longitude 1490972 non-null float64  2 brightness 1490972 non-null float64  3 scan 1490972 non-null float64  4 track 1490972 non-null float64  5 acq\_date 1490972 non-null datetime64[ns]  6 acq\_time 1490972 non-null int64  7 satellite 1490972 non-null object  8 instrument 1490972 non-null object  9 confidence 1490972 non-null int64  10 version 1490972 non-null float64  11 bright\_t31 1490972 non-null float64  12 frp 1490972 non-null float64  13 daynight 1490972 non-null object  14 type 1490972 non-null int64  dtypes: datetime64[ns](1), float64(8), int64(3), object(3)  memory usage: 170.6+ MB |

|  |
| --- |
| # Menghitung jumlah titik panas berdasarkan tanggal  dataset = dataset.groupby(['acq\_date']).size().reset\_index(name='hotspot')  print(dataset) |
| acq\_date hotspot  0 2001-01-01 4  1 2001-01-02 1  2 2001-01-03 2  3 2001-01-04 1  4 2001-01-05 1  ... ... ...  8301 2023-12-27 34  8302 2023-12-28 32  8303 2023-12-29 44  8304 2023-12-30 17  8305 2023-12-31 25  [8306 rows x 2 columns] |

|  |
| --- |
| # create frame  fig, ax = plt.subplots(figsize = (12,4))    # time series plot  ax.plot(    dataset["acq\_date"], dataset["hotspot"],    color="tab:blue", label="Hotspot at 2001 - 2023", linewidth=2  )    # set label-labels  ax.set\_title("Visualization of Hotspot Indonesia",fontsize=14)  ax.set\_xlabel("",fontsize=12)  ax.set\_ylabel("",fontsize=12)  ax.legend(loc="best")  ax.grid(True)    # return values  plt.show() |
|  |
| Gambar x. Output program |
|  |

**Selesai, Selamat Mencoba :3**