9d1t5x2va

April 24, 2025

[6]:
```
import pandas as pd
import matplotlib.pylab as plt
import numpy as np
from sklearn.preprocessing import MinMaxScaler
```

[8]:
```
df = pd.read_csv("iris.csv")
```

checking the contents of the dataset using df.head() and df.tail() functions

[11]:
```
df.head(5)
```

[11]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

[27]:
```
df.tail(5)
```

[27]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	\
145	146	6.7	3.0	5.2	2.3	
146	147	6.3	2.5	5.0	1.9	
147	148	6.5	3.0	5.2	2.0	
148	149	6.2	3.4	5.4	2.3	
149	150	5.9	3.0	5.1	1.8	

	Species
145	Iris-virginica
146	Iris-virginica
147	Iris-virginica
148	Iris-virginica
149	Iris-virginica

[29]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
```

```
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             150 non-null    int64
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

[31]:
```
df.describe()
```

[31]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

Data Processing

[34]:
```
df.isnull()
```

[34]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	False	False	False	False
3	False	False	False	False	False	False
4	False	False	False	False	False	False
..	…	…	…	…	…	…
145	False	False	False	False	False	False
146	False	False	False	False	False	False
147	False	False	False	False	False	False
148	False	False	False	False	False	False
149	False	False	False	False	False	False

[150 rows x 6 columns]

[79]:
```
df.isnull().sum()
```

[79]:
```
Id               0
SepalLengthCm    0
SepalWidthCm     0
```

```
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

[81]:
```
df.notnull()
```

[81]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	True	True	True	True	True	True
1	True	True	True	True	True	True
2	True	True	True	True	True	True
3	True	True	True	True	True	True
4	True	True	True	True	True	True
..	…	…	…	…	…	…
145	True	True	True	True	True	True
146	True	True	True	True	True	True
147	True	True	True	True	True	True
148	True	True	True	True	True	True
149	True	True	True	True	True	True

[150 rows x 6 columns]

[83]:
```
df.describe()
```

[83]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

[85]:
```
# Check dimensions of the dataset
print("Shape of the dataset:", df.shape)
```

```
Shape of the dataset: (150, 6)
```

[87]:
```
# Variable Descriptions
print("\nColumns:")
print(df.columns.tolist())
```

```
Columns:
['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
'Species']
```

Data Formatting and Data Normalization

[90]:
```
# Check data types
df.dtypes
```

[90]:
```
Id                int64
SepalLengthCm    float64
SepalWidthCm     float64
PetalLengthCm    float64
PetalWidthCm     float64
Species           object
dtype: object
```

[92]:
```
print(df.columns)
```

```
Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

[94]:
```
# Convert species column to categorical (for efficiency)
df['Species'] = df['Species'].astype('category')
```

Data Normalization

[99]:
```
scaler = MinMaxScaler()
df[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']] = scaler.
 ↪fit_transform(
    df[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]
)
```

[105]:
```
df.head()

#after normalization the data is scaled between 0 and 1
# Example before and after:
#| SepalLength | SepalWidth | (Before) | |-------------|-------------| | 5.1 |␣
 ↪3.5 | | 7.0 | 3.2 | | 4.6 | 3.1 |

#→ after MinMaxScaler (scales to 0-1 range):

#| SepalLength | SepalWidth | (After) | |-------------|-------------| | 0.38 |␣
 ↪0.75 | | 1.00 | 0.50 | | 0.20 | 0.45 |
# Machine learning models work better when features are on the same scale. and␣
 ↪makes training faster.
```

[105]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	0.222222	0.625000	0.067797	0.041667	Iris-setosa
1	2	0.166667	0.416667	0.067797	0.041667	Iris-setosa
2	3	0.111111	0.500000	0.050847	0.041667	Iris-setosa

```
   3   4       0.083333      0.458333       0.084746      0.041667  Iris-setosa
   4   5       0.194444      0.666667       0.067797      0.041667  Iris-setosa
```

[ ]: