normalization

March 20, 2025

[1]:
```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler , StandardScaler
from sklearn.preprocessing import LabelEncoder , OneHotEncoder
```

[2]:
```
df = pd.read_csv('iris_data.csv')
df.head()
```

[2]:
```
   sepal_length  sepal_width  petal_length  petal_width species
0           5.1          3.5           1.4          0.2  setosa
1           4.9          3.0           1.4          0.2  setosa
2           4.7          3.2           1.3          0.2  setosa
3           4.6          3.1           1.5          0.2  setosa
4           5.0          3.6           1.4          0.2  setosa
```

[4]:
```
df_num = df.drop('species',axis = 1) #axis = 1 drops the columns , axis = 0␣
 ↪drops the row
df_num
```

[4]:
```
     sepal_length  sepal_width  petal_length  petal_width
0             5.1          3.5           1.4          0.2
1             4.9          3.0           1.4          0.2
2             4.7          3.2           1.3          0.2
3             4.6          3.1           1.5          0.2
4             5.0          3.6           1.4          0.2
..            …            …             …            …
145           6.7          3.0           5.2          2.3
146           6.3          2.5           5.0          1.9
147           6.5          3.0           5.2          2.0
148           6.2          3.4           5.4          2.3
149           5.9          3.0           5.1          1.8

[150 rows x 4 columns]
```

[5]:
```
#Normalize the data (MinMax Scaling)
minmax = MinMaxScaler()
df_normalized = pd.DataFrame(minmax.fit_transform(df_num),columns=df_num.
 ↪columns)
```

```
df_normalized.head()
```

[5]:

```
   sepal_length  sepal_width  petal_length  petal_width
0      0.222222     0.625000      0.067797     0.041667
1      0.166667     0.416667      0.067797     0.041667
2      0.111111     0.500000      0.050847     0.041667
3      0.083333     0.458333      0.084746     0.041667
4      0.194444     0.666667      0.067797     0.041667
```

[6]:

```
#Standardize the data (Z-score Scaling)
s = StandardScaler()
df_strandadized = pd.DataFrame(s.fit_transform(df_num),columns = df_num.columns)
df_strandadized.head()
```

[6]:

```
   sepal_length  sepal_width  petal_length  petal_width
0     -0.900681     1.032057     -1.341272    -1.312977
1     -1.143017    -0.124958     -1.341272    -1.312977
2     -1.385353     0.337848     -1.398138    -1.312977
3     -1.506521     0.106445     -1.284407    -1.312977
4     -1.021849     1.263460     -1.341272    -1.312977
```

[7]:

```
#Label Enocder
l = LabelEncoder()
df['species_encoded'] = l.fit_transform(df['species'])
df.head()

result = df.drop('species',axis = 1)
result
```

[7]:

```
     sepal_length  sepal_width  petal_length  petal_width  species_encoded
0             5.1          3.5           1.4          0.2                0
1             4.9          3.0           1.4          0.2                0
2             4.7          3.2           1.3          0.2                0
3             4.6          3.1           1.5          0.2                0
4             5.0          3.6           1.4          0.2                0
..            …            …             …            …              …
145           6.7          3.0           5.2          2.3                2
146           6.3          2.5           5.0          1.9                2
147           6.5          3.0           5.2          2.0                2
148           6.2          3.4           5.4          2.3                2
149           5.9          3.0           5.1          1.8                2

[150 rows x 5 columns]
```

[8]:

```
#Reloading the data
df = pd.read_csv('iris_data.csv')
df.head()
```

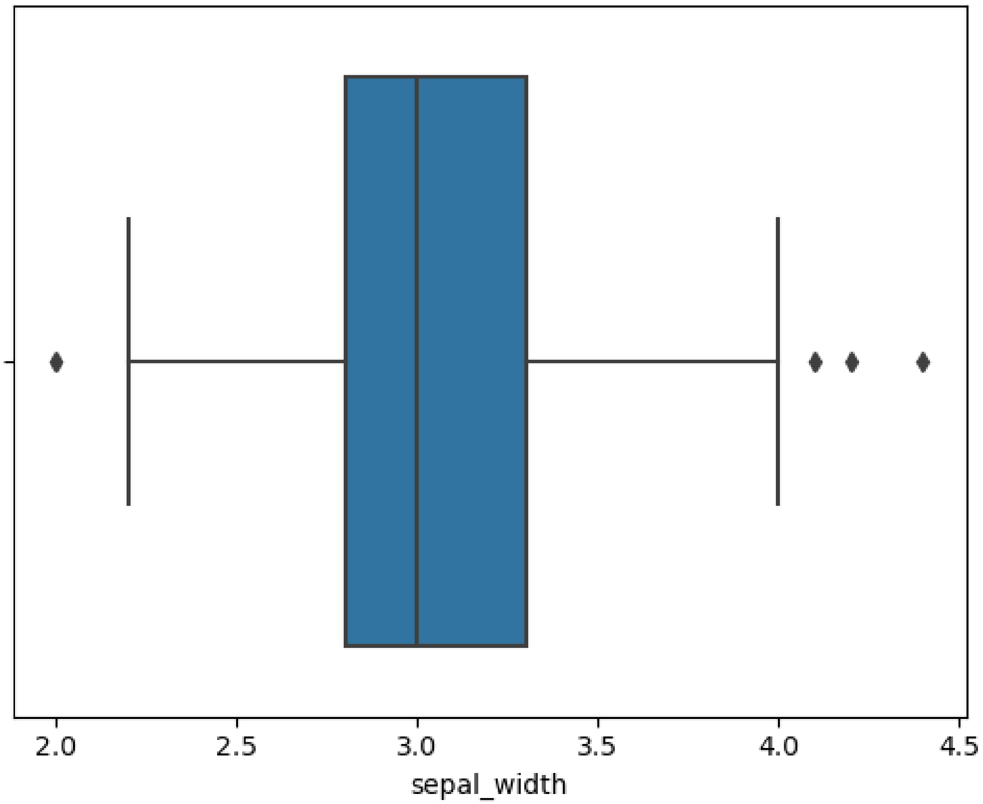
```
[8]:    sepal_length  sepal_width  petal_length  petal_width species
     0           5.1          3.5           1.4          0.2  setosa
     1           4.9          3.0           1.4          0.2  setosa
     2           4.7          3.2           1.3          0.2  setosa
     3           4.6          3.1           1.5          0.2  setosa
     4           5.0          3.6           1.4          0.2  setosa
```

```
[10]: #One Hot Encoder
      onehot_encoder = OneHotEncoder(sparse_output = False)
      encoded_species = pd.DataFrame(onehot_encoder.fit_transform(df[['species']]))
      encoded_species

      merged = pd.concat([df,encoded_species],axis = 1)
      merged

      result = merged.drop('species',axis = 1)
      result
```

```
[10]:      sepal_length  sepal_width  petal_length  petal_width    0    1    2
      0             5.1          3.5           1.4          0.2  1.0  0.0  0.0
      1             4.9          3.0           1.4          0.2  1.0  0.0  0.0
      2             4.7          3.2           1.3          0.2  1.0  0.0  0.0
      3             4.6          3.1           1.5          0.2  1.0  0.0  0.0
      4             5.0          3.6           1.4          0.2  1.0  0.0  0.0
      ..              …            …             …          …    …    …    …
      145           6.7          3.0           5.2          2.3  0.0  0.0  1.0
      146           6.3          2.5           5.0          1.9  0.0  0.0  1.0
      147           6.5          3.0           5.2          2.0  0.0  0.0  1.0
      148           6.2          3.4           5.4          2.3  0.0  0.0  1.0
      149           5.9          3.0           5.1          1.8  0.0  0.0  1.0

      [150 rows x 7 columns]
```

```
[11]: import seaborn as sns
      sns.boxplot(x='sepal_width',data=df)
```

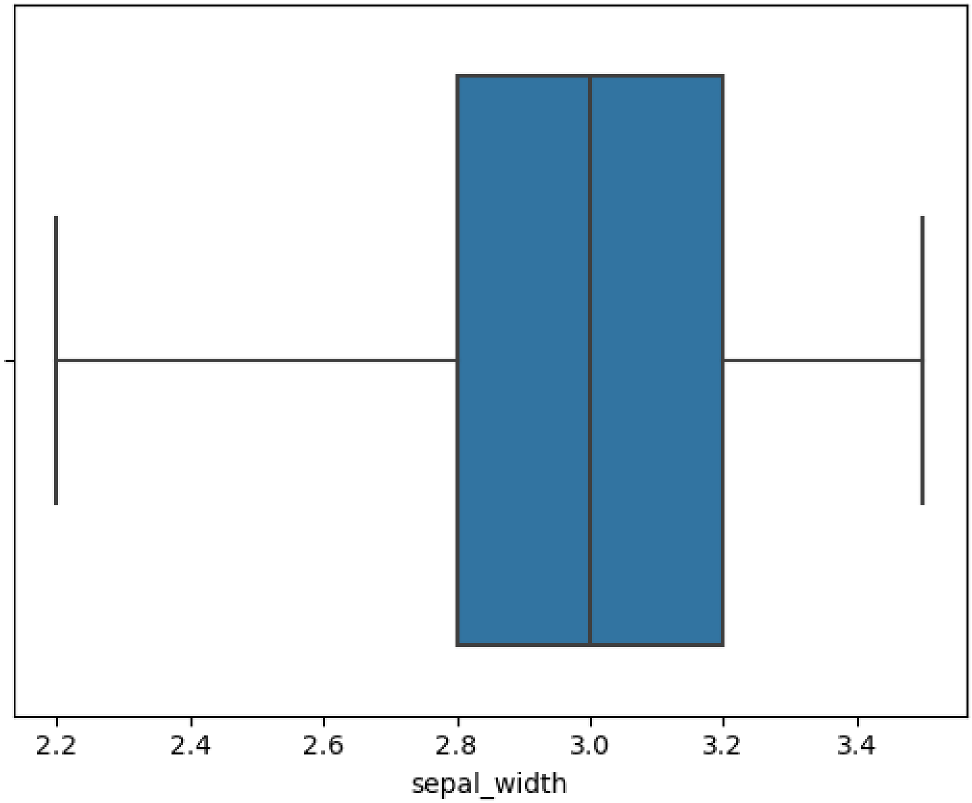
```
[11]: <Axes: xlabel='sepal_width'>
```



[12]:
```
q1 = df['sepal_width'].quantile(0.25)
q3 = df['sepal_width'].quantile(0.75)
iqr = q3-q1       #interquartile range
lower_bound = q1 - 1.5*iqr
upper_bound = q1 + 1.5*iqr
df = df[(df['sepal_width']>lower_bound)&(df['sepal_width']<upper_bound)]
```

[13]:
```
sns.boxplot(x='sepal_width',data = df)
```

[13]:
```
<Axes: xlabel='sepal_width'>
```


2.2
2.4
2.6
2.8
3.0
3.2
3.4
sepal_width

[ ]: