wtech
Teknolojide Kadın Derneği

Introduction to Machine Learning with Python

Unsupervised Learning

Dr. Süha Tuna
İTÜ Informatics Institute

Unsupervised Learning Types

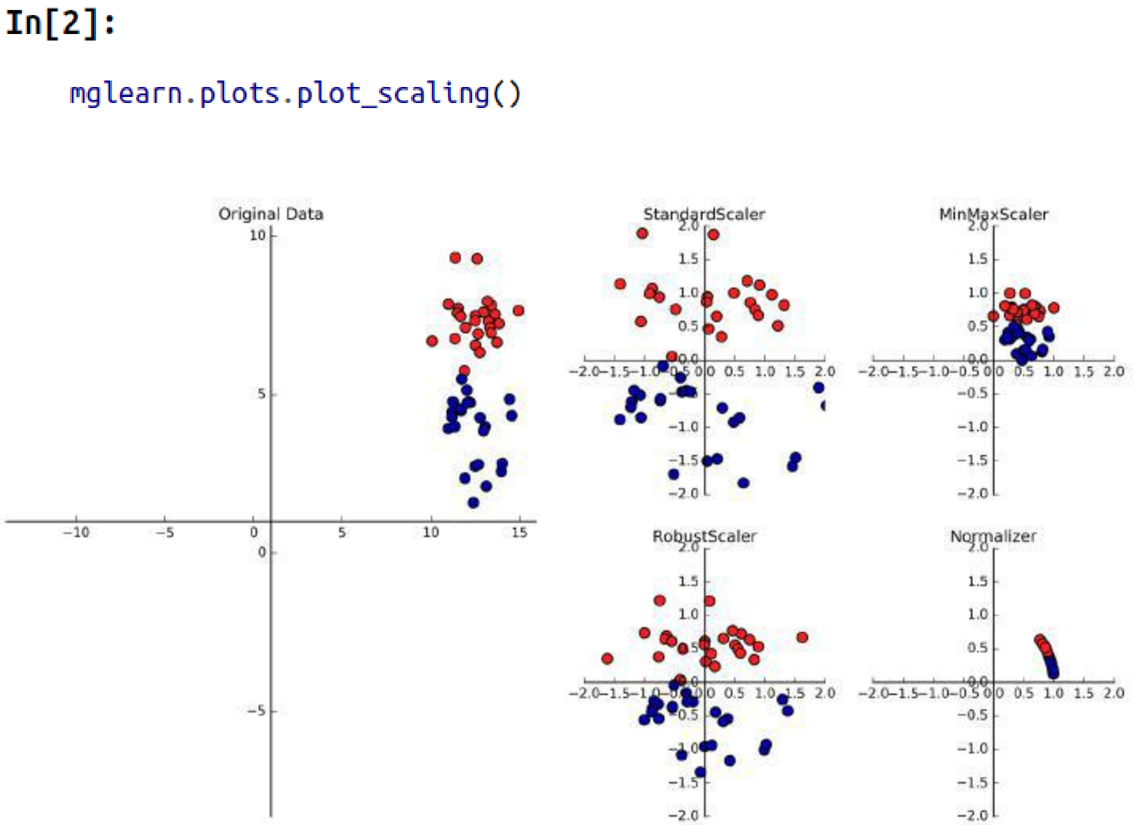
Unsupervised Transformations

Clustering

Challanges

- **Key Challenge: Evaluation**
  - No label information → No clear 'correct' output
  - Hard to determine if algorithm 'did well'
  - Example: Clustering by profile vs. frontal view may not match user intent
- **Manual Inspection is Common**
  - Often the only way to evaluate results
  - Requires domain knowledge
  - Unsupervised results are subjective
- **Primary Use Cases**
  - Exploratory data analysis
  - Preprocessing for supervised learning
  - Improve accuracy or efficiency with learned representations
- **Preprocessing Methods**
  - Used in both supervised and unsupervised contexts
  - Examples: Scaling, normalization
  - Do not use label information → inherently unsupervised

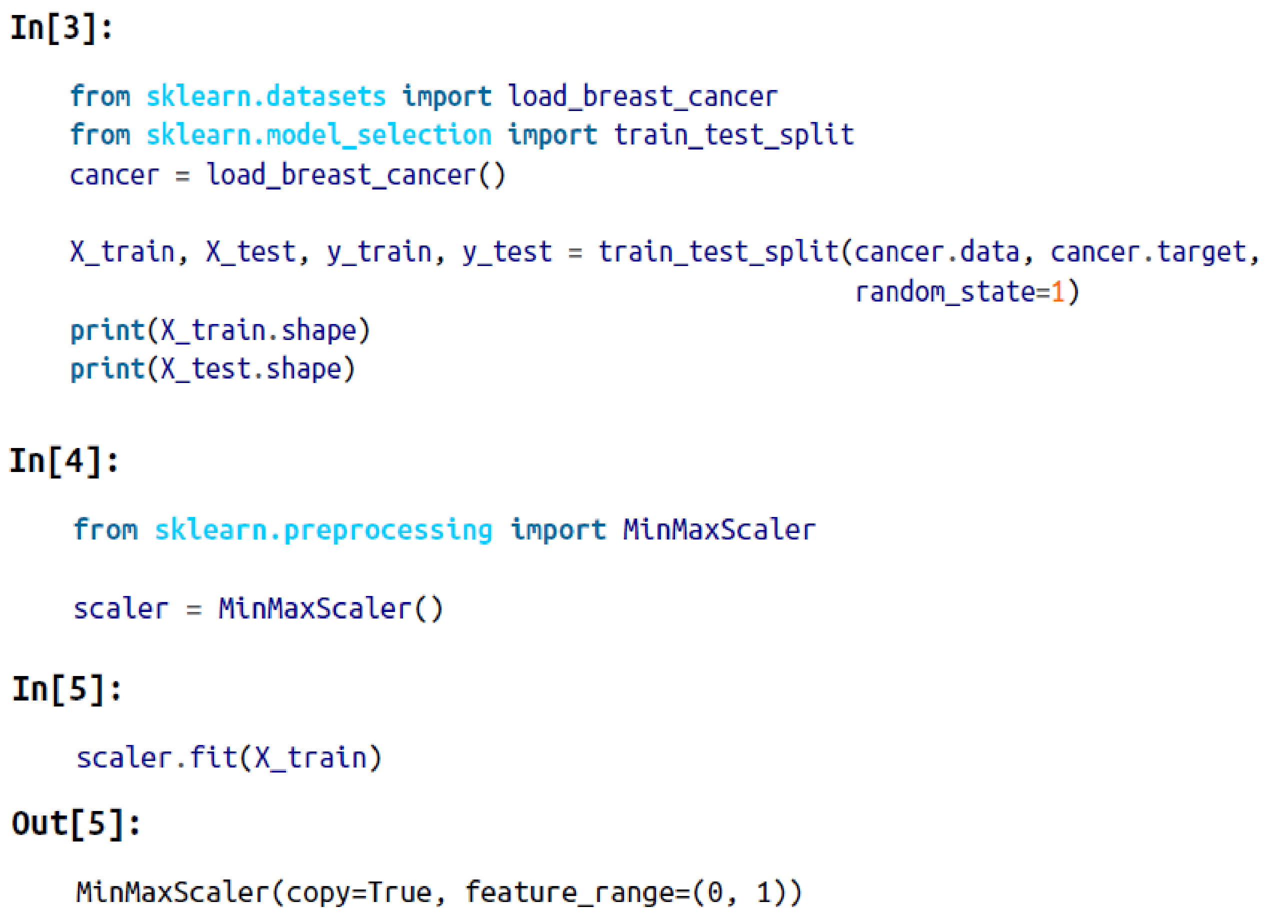
Preprocessing and Scaling



```
In[2]:

    mglearn.plots.plot_scaling()
```

- StandardScaler
  - Centers each feature (mean = 0, variance = 1)
  - Does NOT guarantee specific min/max values
  - Useful for models assuming Gaussian distribution
- RobustScaler
  - Uses median and IQR (interquartile range)
  - Resistant to outliers and noisy measurements
  - Better for skewed or corrupted data
- MinMaxScaler
  - Scales all features to the [0, 1] range
  - Useful for algorithms requiring bounded input (e.g., neural networks)
  - Sensitive to outliers
- Normalizer
  - Scales each sample to unit norm (length = 1)
  - Preserves direction, not magnitude
  - Useful when angle/direction matters (e.g., text, cosine similarity)

Applying Data Transformations



In[3]:

```
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
cancer = load_breast_cancer()

X_train, X_test, y_train, y_test = train_test_split(cancer.data, cancer.target,
                                                    random_state=1)
print(X_train.shape)
print(X_test.shape)
```

In[4]:

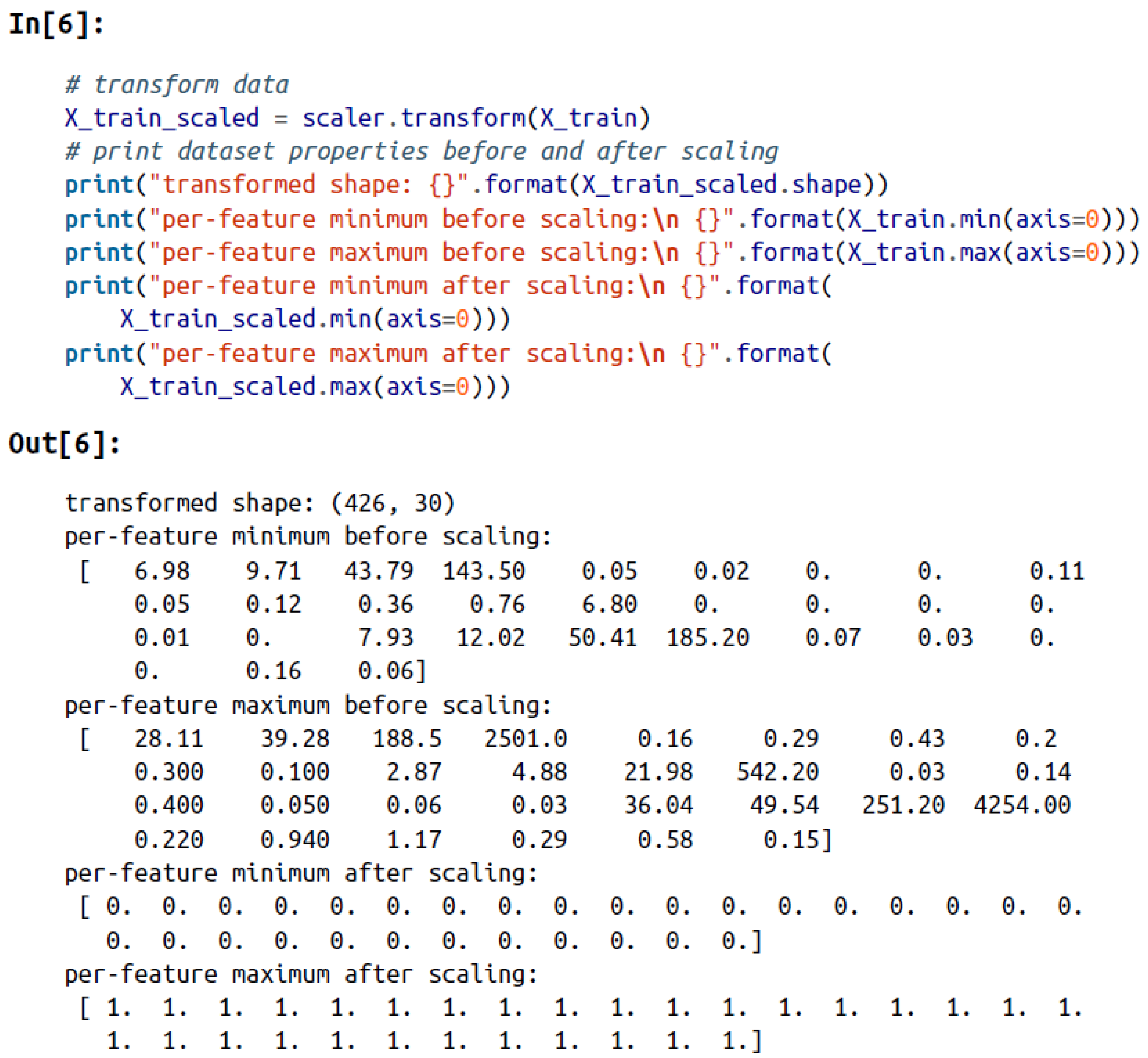
```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
```

In[5]:

```
scaler.fit(X_train)
```

Out[5]:

```
MinMaxScaler(copy=True, feature_range=(0, 1))
```



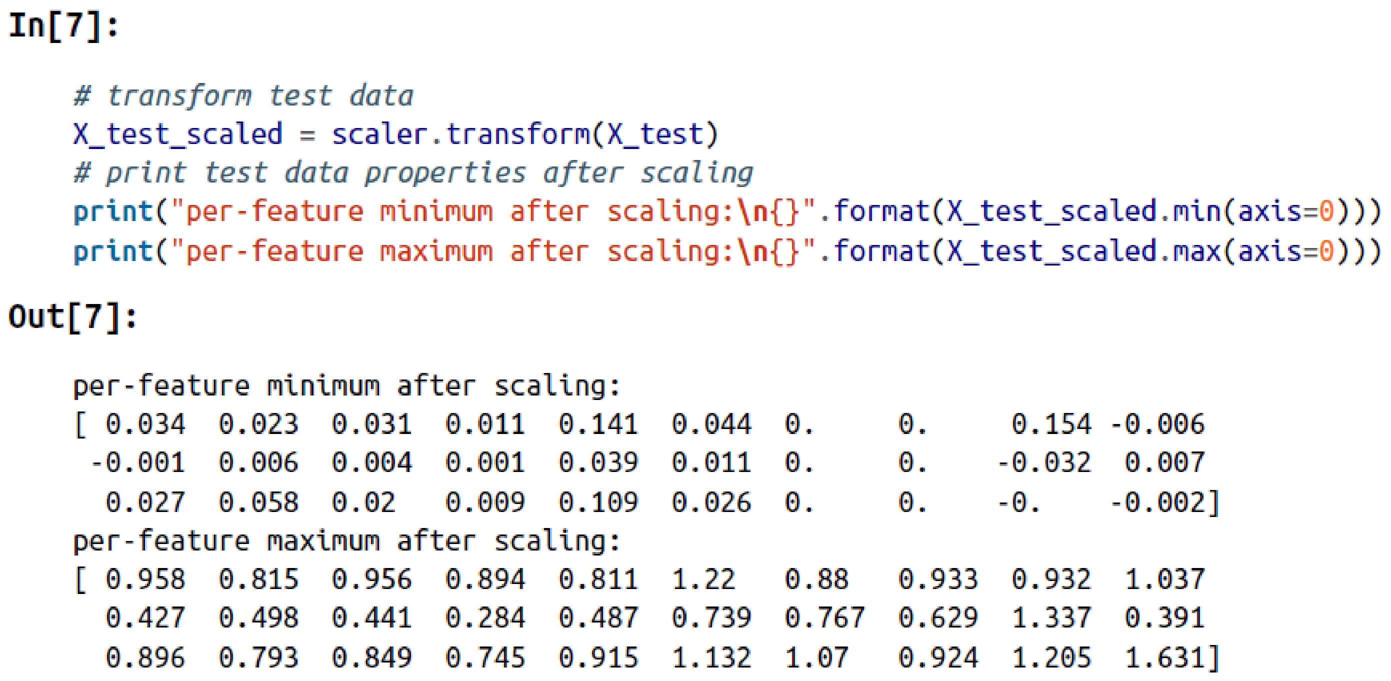
In[6]:

```
# transform data
X_train_scaled = scaler.transform(X_train)
# print dataset properties before and after scaling
print("transformed shape: {}".format(X_train_scaled.shape))
print("per-feature minimum before scaling:\n {}".format(X_train.min(axis=0)))
print("per-feature maximum before scaling:\n {}".format(X_train.max(axis=0)))
print("per-feature minimum after scaling:\n {}".format(
    X_train_scaled.min(axis=0)))
print("per-feature maximum after scaling:\n {}".format(
    X_train_scaled.max(axis=0)))
```

Out[6]:

```
transformed shape: (426, 30)
per-feature minimum before scaling:
 [   6.98    9.71   43.79  143.50    0.05    0.02    0.      0.      0.11
     0.05    0.12    0.36    0.76    6.80    0.      0.      0.      0.
     0.01    0.      7.93   12.02   50.41  185.20    0.07    0.03    0.
     0.      0.16    0.06]
per-feature maximum before scaling:
 [   28.11    39.28   188.5   2501.0     0.16     0.29     0.43     0.2
     0.300    0.100    2.87     4.88    21.98   542.20     0.03     0.14
     0.400    0.050    0.06     0.03    36.04    49.54   251.20  4254.00
     0.220    0.940    1.17     0.29     0.58     0.15]
per-feature minimum after scaling:
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.
   0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
per-feature maximum after scaling:
 [ 1.  1.  1.  1.  1.  1.  1.  1.  1.  1.  1.  1.  1.  1.  1.  1.  1.  1.
   1.  1.  1.  1.  1.  1.  1.  1.  1.  1.  1.  1.]
```

Applying Data Transformations



In[7]:

```
# transform test data
X_test_scaled = scaler.transform(X_test)
# print test data properties after scaling
print("per-feature minimum after scaling:\n{}".format(X_test_scaled.min(axis=0)))
print("per-feature maximum after scaling:\n{}".format(X_test_scaled.max(axis=0)))
```

Out[7]:

```
per-feature minimum after scaling:
[ 0.034  0.023  0.031  0.011  0.141  0.044  0.     0.     0.154 -0.006
 -0.001  0.006  0.004  0.001  0.039  0.011  0.     0.    -0.032  0.007
  0.027  0.058  0.02   0.009  0.109  0.026  0.     0.    -0.    -0.002]
per-feature maximum after scaling:
[ 0.958  0.815  0.956  0.894  0.811  1.22   0.88   0.933  0.932  1.037
  0.427  0.498  0.441  0.284  0.487  0.739  0.767  0.629  1.337  0.391
  0.896  0.793  0.849  0.745  0.915  1.132  1.07   0.924  1.205  1.631]
```