SVM

July 24, 2020

[1]:
```
import mglearn
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.svm import LinearSVC
from sklearn.datasets import load_iris
```

[2]:
```
iris = load_iris()
```

[3]:
```
iris
```

[3]:
```
{'data': array([[5.1, 3.5, 1.4, 0.2],
        [4.9, 3. , 1.4, 0.2],
        [4.7, 3.2, 1.3, 0.2],
        [4.6, 3.1, 1.5, 0.2],
        [5. , 3.6, 1.4, 0.2],
        [5.4, 3.9, 1.7, 0.4],
        [4.6, 3.4, 1.4, 0.3],
        [5. , 3.4, 1.5, 0.2],
        [4.4, 2.9, 1.4, 0.2],
        [4.9, 3.1, 1.5, 0.1],
        [5.4, 3.7, 1.5, 0.2],
        [4.8, 3.4, 1.6, 0.2],
        [4.8, 3. , 1.4, 0.1],
        [4.3, 3. , 1.1, 0.1],
        [5.8, 4. , 1.2, 0.2],
        [5.7, 4.4, 1.5, 0.4],
        [5.4, 3.9, 1.3, 0.4],
        [5.1, 3.5, 1.4, 0.3],
        [5.7, 3.8, 1.7, 0.3],
        [5.1, 3.8, 1.5, 0.3],
        [5.4, 3.4, 1.7, 0.2],
        [5.1, 3.7, 1.5, 0.4],
        [4.6, 3.6, 1. , 0.2],
        [5.1, 3.3, 1.7, 0.5],
        [4.8, 3.4, 1.9, 0.2],
        [5. , 3. , 1.6, 0.2],
        [5. , 3.4, 1.6, 0.4],
```

```
[5.2, 3.5, 1.5, 0.2],
[5.2, 3.4, 1.4, 0.2],
[4.7, 3.2, 1.6, 0.2],
[4.8, 3.1, 1.6, 0.2],
[5.4, 3.4, 1.5, 0.4],
[5.2, 4.1, 1.5, 0.1],
[5.5, 4.2, 1.4, 0.2],
[4.9, 3.1, 1.5, 0.2],
[5. , 3.2, 1.2, 0.2],
[5.5, 3.5, 1.3, 0.2],
[4.9, 3.6, 1.4, 0.1],
[4.4, 3. , 1.3, 0.2],
[5.1, 3.4, 1.5, 0.2],
[5. , 3.5, 1.3, 0.3],
[4.5, 2.3, 1.3, 0.3],
[4.4, 3.2, 1.3, 0.2],
[5. , 3.5, 1.6, 0.6],
[5.1, 3.8, 1.9, 0.4],
[4.8, 3. , 1.4, 0.3],
[5.1, 3.8, 1.6, 0.2],
[4.6, 3.2, 1.4, 0.2],
[5.3, 3.7, 1.5, 0.2],
[5. , 3.3, 1.4, 0.2],
[7. , 3.2, 4.7, 1.4],
[6.4, 3.2, 4.5, 1.5],
[6.9, 3.1, 4.9, 1.5],
[5.5, 2.3, 4. , 1.3],
[6.5, 2.8, 4.6, 1.5],
[5.7, 2.8, 4.5, 1.3],
[6.3, 3.3, 4.7, 1.6],
[4.9, 2.4, 3.3, 1. ],
[6.6, 2.9, 4.6, 1.3],
[5.2, 2.7, 3.9, 1.4],
[5. , 2. , 3.5, 1. ],
[5.9, 3. , 4.2, 1.5],
[6. , 2.2, 4. , 1. ],
[6.1, 2.9, 4.7, 1.4],
[5.6, 2.9, 3.6, 1.3],
[6.7, 3.1, 4.4, 1.4],
[5.6, 3. , 4.5, 1.5],
[5.8, 2.7, 4.1, 1. ],
[6.2, 2.2, 4.5, 1.5],
[5.6, 2.5, 3.9, 1.1],
[5.9, 3.2, 4.8, 1.8],
[6.1, 2.8, 4. , 1.3],
[6.3, 2.5, 4.9, 1.5],
[6.1, 2.8, 4.7, 1.2],
```

```
[6.4, 2.9, 4.3, 1.3],
[6.6, 3. , 4.4, 1.4],
[6.8, 2.8, 4.8, 1.4],
[6.7, 3. , 5. , 1.7],
[6. , 2.9, 4.5, 1.5],
[5.7, 2.6, 3.5, 1. ],
[5.5, 2.4, 3.8, 1.1],
[5.5, 2.4, 3.7, 1. ],
[5.8, 2.7, 3.9, 1.2],
[6. , 2.7, 5.1, 1.6],
[5.4, 3. , 4.5, 1.5],
[6. , 3.4, 4.5, 1.6],
[6.7, 3.1, 4.7, 1.5],
[6.3, 2.3, 4.4, 1.3],
[5.6, 3. , 4.1, 1.3],
[5.5, 2.5, 4. , 1.3],
[5.5, 2.6, 4.4, 1.2],
[6.1, 3. , 4.6, 1.4],
[5.8, 2.6, 4. , 1.2],
[5. , 2.3, 3.3, 1. ],
[5.6, 2.7, 4.2, 1.3],
[5.7, 3. , 4.2, 1.2],
[5.7, 2.9, 4.2, 1.3],
[6.2, 2.9, 4.3, 1.3],
[5.1, 2.5, 3. , 1.1],
[5.7, 2.8, 4.1, 1.3],
[6.3, 3.3, 6. , 2.5],
[5.8, 2.7, 5.1, 1.9],
[7.1, 3. , 5.9, 2.1],
[6.3, 2.9, 5.6, 1.8],
[6.5, 3. , 5.8, 2.2],
[7.6, 3. , 6.6, 2.1],
[4.9, 2.5, 4.5, 1.7],
[7.3, 2.9, 6.3, 1.8],
[6.7, 2.5, 5.8, 1.8],
[7.2, 3.6, 6.1, 2.5],
[6.5, 3.2, 5.1, 2. ],
[6.4, 2.7, 5.3, 1.9],
[6.8, 3. , 5.5, 2.1],
[5.7, 2.5, 5. , 2. ],
[5.8, 2.8, 5.1, 2.4],
[6.4, 3.2, 5.3, 2.3],
[6.5, 3. , 5.5, 1.8],
[7.7, 3.8, 6.7, 2.2],
[7.7, 2.6, 6.9, 2.3],
[6. , 2.2, 5. , 1.5],
[6.9, 3.2, 5.7, 2.3],
```

```
        [5.6, 2.8, 4.9, 2. ],
        [7.7, 2.8, 6.7, 2. ],
        [6.3, 2.7, 4.9, 1.8],
        [6.7, 3.3, 5.7, 2.1],
        [7.2, 3.2, 6. , 1.8],
        [6.2, 2.8, 4.8, 1.8],
        [6.1, 3. , 4.9, 1.8],
        [6.4, 2.8, 5.6, 2.1],
        [7.2, 3. , 5.8, 1.6],
        [7.4, 2.8, 6.1, 1.9],
        [7.9, 3.8, 6.4, 2. ],
        [6.4, 2.8, 5.6, 2.2],
        [6.3, 2.8, 5.1, 1.5],
        [6.1, 2.6, 5.6, 1.4],
        [7.7, 3. , 6.1, 2.3],
        [6.3, 3.4, 5.6, 2.4],
        [6.4, 3.1, 5.5, 1.8],
        [6. , 3. , 4.8, 1.8],
        [6.9, 3.1, 5.4, 2.1],
        [6.7, 3.1, 5.6, 2.4],
        [6.9, 3.1, 5.1, 2.3],
        [5.8, 2.7, 5.1, 1.9],
        [6.8, 3.2, 5.9, 2.3],
        [6.7, 3.3, 5.7, 2.5],
        [6.7, 3. , 5.2, 2.3],
        [6.3, 2.5, 5. , 1.9],
        [6.5, 3. , 5.2, 2. ],
        [6.2, 3.4, 5.4, 2.3],
        [5.9, 3. , 5.1, 1.8]]),
 'target': array([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, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 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, 1, 1, 1, 1, 1, 1, 1, 1,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
        2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
        2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]),
 'target_names': array(['setosa', 'versicolor', 'virginica'], dtype='<U10'),
 'DESCR': '.. _iris_dataset:\n\nIris plants
dataset\n--------------------\n\n**Data Set Characteristics:**\n\n    :Number of
Instances: 150 (50 in each of three classes)\n    :Number of Attributes: 4
numeric, predictive attributes and the class\n    :Attribute Information:\n
- sepal length in cm\n        - sepal width in cm\n        - petal length in
cm\n        - petal width in cm\n        - class:\n                - Iris-
Setosa\n                - Iris-Versicolour\n                - Iris-Virginica\n
\n    :Summary Statistics:\n\n    ============== ==== ==== ======= =====
====================\n                    Min  Max   Mean    SD   Class
```

```
Correlation\n    ============== ==== ==== ======= ===== ====================\n
sepal length:   4.3  7.9   5.84   0.83    0.7826\n    sepal width:    2.0  4.4
3.05   0.43   -0.4194\n    petal length:   1.0  6.9   3.76   1.76    0.9490
(high!)\n    petal width:    0.1  2.5   1.20   0.76    0.9565  (high!)\n
============== ==== ==== ======= ===== ====================\n\n    :Missing
Attribute Values: None\n    :Class Distribution: 33.3% for each of 3 classes.\n
:Creator: R.A. Fisher\n    :Donor: Michael Marshall
(MARSHALL%PLU@io.arc.nasa.gov)\n    :Date: July, 1988\n\nThe famous Iris
database, first used by Sir R.A. Fisher. The dataset is taken\nfrom Fisher\'s
paper. Note that it\'s the same as in R, but not as in the UCI\nMachine Learning
Repository, which has two wrong data points.\n\nThis is perhaps the best known
database to be found in the\npattern recognition literature.  Fisher\'s paper is
a classic in the field and\nis referenced frequently to this day.  (See Duda &
Hart, for example.)  The\ndata set contains 3 classes of 50 instances each,
where each class refers to a\ntype of iris plant.  One class is linearly
separable from the other 2; the\nlatter are NOT linearly separable from each
other.\n\n.. topic:: References\n\n   - Fisher, R.A. "The use of multiple
measurements in taxonomic problems"\n     Annual Eugenics, 7, Part II, 179-188
(1936); also in "Contributions to\n     Mathematical Statistics" (John Wiley,
NY, 1950).\n   - Duda, R.O., & Hart, P.E. (1973) Pattern Classification and
Scene Analysis.\n     (Q327.D83) John Wiley & Sons.  ISBN 0-471-22361-1.  See
page 218.\n   - Dasarathy, B.V. (1980) "Nosing Around the Neighborhood: A New
System\n     Structure and Classification Rule for Recognition in Partially
Exposed\n     Environments".  IEEE Transactions on Pattern Analysis and
Machine\n     Intelligence, Vol. PAMI-2, No. 1, 67-71.\n   - Gates, G.W. (1972)
"The Reduced Nearest Neighbor Rule".  IEEE Transactions\n     on Information
Theory, May 1972, 431-433.\n   - See also: 1988 MLC Proceedings, 54-64.
Cheeseman et al"s AUTOCLASS II\n     conceptual clustering system finds 3
classes in the data.\n   - Many, many more …',
 'feature_names': ['sepal length (cm)',
  'sepal width (cm)',
  'petal length (cm)',
  'petal width (cm)'],
 'filename': '/opt/conda/lib/python3.7/site-
packages/sklearn/datasets/data/iris.csv'}
```

[4]:
```
X = iris.data[50:, 2:]
y = iris.target[50:] - 1
```

[5]:
```
X
```

[5]:
```
array([[4.7, 1.4],
       [4.5, 1.5],
       [4.9, 1.5],
       [4. , 1.3],
       [4.6, 1.5],
       [4.5, 1.3],
```

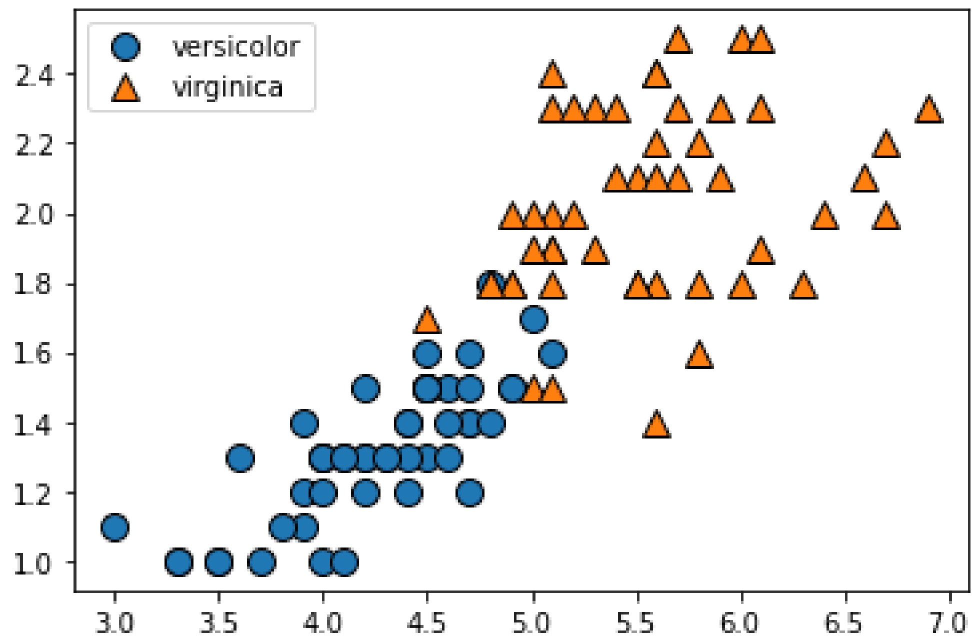
```
[4.7, 1.6],
[3.3, 1. ],
[4.6, 1.3],
[3.9, 1.4],
[3.5, 1. ],
[4.2, 1.5],
[4. , 1. ],
[4.7, 1.4],
[3.6, 1.3],
[4.4, 1.4],
[4.5, 1.5],
[4.1, 1. ],
[4.5, 1.5],
[3.9, 1.1],
[4.8, 1.8],
[4. , 1.3],
[4.9, 1.5],
[4.7, 1.2],
[4.3, 1.3],
[4.4, 1.4],
[4.8, 1.4],
[5. , 1.7],
[4.5, 1.5],
[3.5, 1. ],
[3.8, 1.1],
[3.7, 1. ],
[3.9, 1.2],
[5.1, 1.6],
[4.5, 1.5],
[4.5, 1.6],
[4.7, 1.5],
[4.4, 1.3],
[4.1, 1.3],
[4. , 1.3],
[4.4, 1.2],
[4.6, 1.4],
[4. , 1.2],
[3.3, 1. ],
[4.2, 1.3],
[4.2, 1.2],
[4.2, 1.3],
[4.3, 1.3],
[3. , 1.1],
[4.1, 1.3],
[6. , 2.5],
[5.1, 1.9],
[5.9, 2.1],
```

```
[5.6, 1.8],
[5.8, 2.2],
[6.6, 2.1],
[4.5, 1.7],
[6.3, 1.8],
[5.8, 1.8],
[6.1, 2.5],
[5.1, 2. ],
[5.3, 1.9],
[5.5, 2.1],
[5. , 2. ],
[5.1, 2.4],
[5.3, 2.3],
[5.5, 1.8],
[6.7, 2.2],
[6.9, 2.3],
[5. , 1.5],
[5.7, 2.3],
[4.9, 2. ],
[6.7, 2. ],
[4.9, 1.8],
[5.7, 2.1],
[6. , 1.8],
[4.8, 1.8],
[4.9, 1.8],
[5.6, 2.1],
[5.8, 1.6],
[6.1, 1.9],
[6.4, 2. ],
[5.6, 2.2],
[5.1, 1.5],
[5.6, 1.4],
[6.1, 2.3],
[5.6, 2.4],
[5.5, 1.8],
[4.8, 1.8],
[5.4, 2.1],
[5.6, 2.4],
[5.1, 2.3],
[5.1, 1.9],
[5.9, 2.3],
[5.7, 2.5],
[5.2, 2.3],
[5. , 1.9],
[5.2, 2. ],
[5.4, 2.3],
[5.1, 1.8]])
```

[6]:
```
y
```

[6]:
```
array([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, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 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, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1])
```

[7]:
```
mglearn.discrete_scatter(X[:, 0], X[:, 1], y)
plt.legend(['versicolor', 'virginica'], loc='best')
plt.show()
```

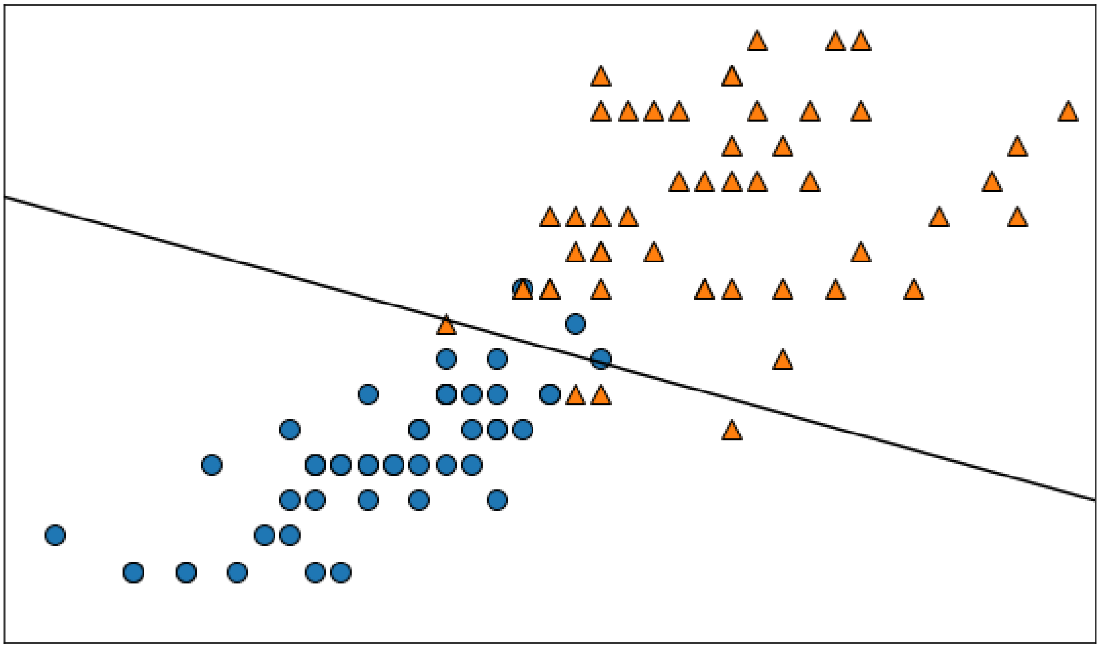


[8]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y,␣
↪random_state=0)
```

[9]:
```
svm = LinearSVC().fit(X_train, y_train)
```

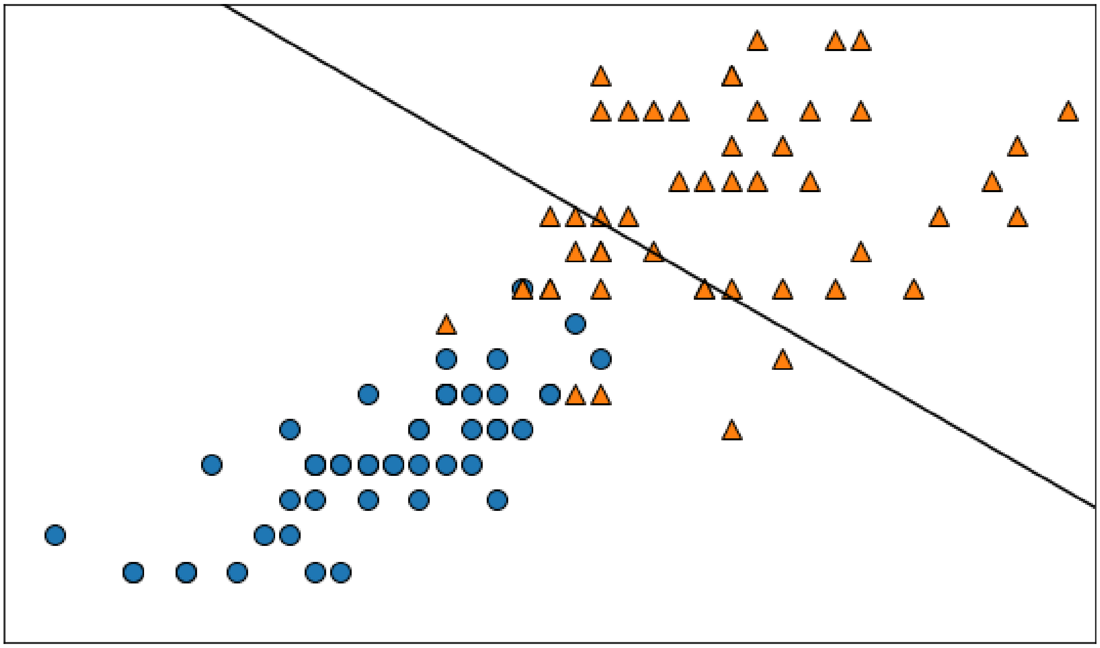
[10]:
```
def plot_separator(model):
    plt.figure(figsize=(10, 6))
    mglearn.plots.plot_2d_separator(model, X)
    mglearn.discrete_scatter(X[:, 0], X[:, 1], y)
    plt.xlabel('petal length')
    plt.xlabel('petal width')
    plt.xlim(2.8, 7.0)
    plt.ylim(0.8, 2.6)
    plt.show()
```

[11]:
```
plot_separator(svm)
```



[12]:
```
svm_100 = LinearSVC(C=100).fit(X_train, y_train)
plot_separator(svm_100)
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/svm/_base.py:947:
ConvergenceWarning: Liblinear failed to converge, increase the number of
iterations.
  "the number of iterations.", ConvergenceWarning)
```



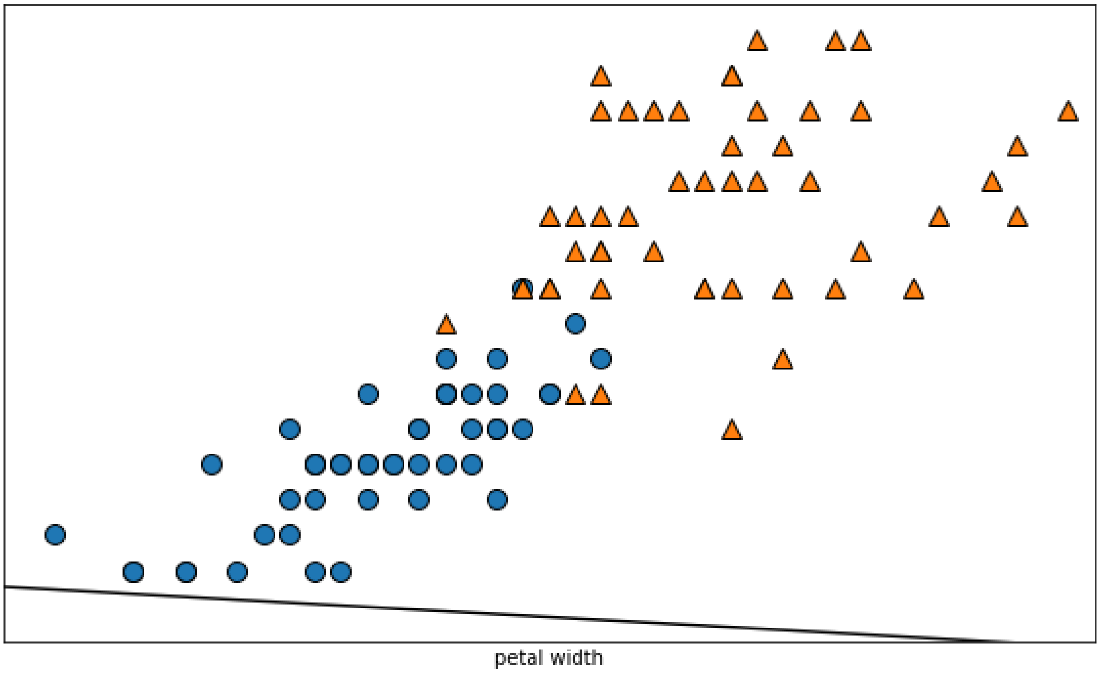
[13]:
```
print('score on training set: {:.2f}'.format(svm.score(X_train, y_train)))
print('score on test set: {:.2f}'.format(svm.score(X_test, y_test)))
```

```
score on training set: 0.91
score on test set: 1.00
```

[14]:
```
print('score on training set: {:.2f}'.format(svm_100.score(X_train, y_train)))
print('score on test set: {:.2f}'.format(svm_100.score(X_test, y_test)))
```

```
score on training set: 0.81
score on test set: 0.88
```

[15]:
```
svm_001 = LinearSVC(C=0.01).fit(X_train, y_train)
plot_separator(svm_001)
```



[16]:
```
print('score on training set: {:.2f}'.format(svm_001.score(X_train, y_train)))
print('score on test set: {:.2f}'.format(svm_001.score(X_test, y_test)))
```

```
score on training set: 0.51
score on test set: 0.48
```

[17]:
```
import mglearn
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_moons
```

[18]:
```
moons = make_moons(n_samples=200, noise=0.1, random_state=0)
moons
```

[18]:
```
(array([[ 7.92357355e-01,  5.02648573e-01],
        [ 1.63158315e+00, -4.63896705e-01],
        [-6.71092674e-02,  2.67767057e-01],
        [-1.04412427e+00, -1.82607610e-01],
        [ 1.76704822e+00, -1.98609868e-01],
        [ 1.90607398e+00, -7.10915927e-02],
        [ 9.62192129e-01,  2.61986075e-01],
        [ 8.86813848e-01, -4.84896235e-01],
        [ 8.68935196e-01,  3.61092776e-01],
        [ 1.15352953e+00, -5.72352929e-01],
        [-3.70714493e-01,  7.21655833e-01],
```

```
[ 1.95523229e-01, -2.64439358e-01],
[ 1.80949658e+00, -1.94707885e-01],
[ 1.29046617e+00, -3.23893778e-01],
[-8.23662620e-01,  7.17643700e-01],
[ 9.85881879e-01,  1.96706218e-01],
[ 5.81943503e-01,  9.06311048e-01],
[ 2.76118902e-01, -2.87451069e-01],
[ 3.17284813e-01,  8.54200996e-01],
[-8.52697952e-01,  9.32383274e-01],
[ 1.97796688e+00,  1.04796611e-02],
[ 8.64249290e-01, -4.63242870e-01],
[ 2.57499053e-01,  1.00362573e+00],
[ 1.28213982e+00, -2.86519496e-01],
[-2.29884751e-01,  1.10576822e+00],
[ 2.29022777e-01,  1.42127683e-02],
[ 1.89450511e+00, -1.02565542e-01],
[ 1.86521084e+00,  1.96753138e-02],
[ 9.76884442e-02, -2.94749068e-02],
[ 1.03579202e-01,  8.07972963e-01],
[ 9.07891105e-01,  5.59120905e-01],
[-2.82483560e-01,  6.97029880e-01],
[ 9.49718195e-01, -4.29281289e-01],
[ 2.44475950e-02,  2.31178102e-01],
[ 7.62407790e-01,  7.01679032e-01],
[ 1.87210764e+00,  2.35600223e-01],
[ 4.77778324e-01, -1.64382571e-01],
[ 9.31783953e-01,  7.18760735e-02],
[-8.12884298e-01,  5.62089952e-01],
[-1.00520615e-01,  4.03688699e-01],
[ 5.82424127e-01, -4.68412597e-01],
[-3.77657209e-01,  8.52782368e-01],
[ 6.28019791e-01,  6.90115928e-01],
[ 1.21605162e-01,  3.53588146e-02],
[ 1.54344475e+00, -3.47674056e-01],
[-3.30952410e-01,  1.00049623e+00],
[ 1.06619086e+00,  1.33052899e-01],
[-7.14739658e-01,  6.19454519e-01],
[ 1.63968968e-01, -1.74798733e-01],
[ 9.33487565e-01, -4.40111162e-01],
[-7.63797985e-01,  8.26528213e-01],
[ 3.79557530e-01, -2.02301374e-01],
[ 2.07209923e+00,  3.52001635e-02],
[-8.11609003e-01,  5.10548387e-02],
[ 1.10617390e+00, -5.00532861e-01],
[ 7.83108552e-01,  6.23681079e-01],
[ 5.19314634e-01,  6.58138927e-01],
[ 1.15369377e+00, -5.06557246e-01],
```

```
[ 2.01456988e-01,  9.30021747e-01],
[-2.10315426e-01,  1.00700943e+00],
[ 2.56648130e-01, -2.48738776e-01],
[-1.03710314e+00,  2.30748836e-01],
[ 1.72148759e+00, -2.08345383e-01],
[ 9.24799914e-01,  4.94247065e-01],
[ 1.08911468e+00, -4.40836789e-01],
[-1.16502510e+00,  2.52779074e-01],
[-1.75082328e-01,  1.04012819e+00],
[ 1.51596993e+00, -3.52163413e-01],
[ 1.08086088e-01,  9.56975505e-01],
[-9.57080149e-01,  5.12567989e-01],
[ 6.47834107e-01, -4.86523627e-01],
[-1.02340159e+00,  5.31509316e-01],
[ 3.13142058e-01, -5.09754017e-01],
[-2.64515467e-01,  5.80672612e-01],
[ 8.04988122e-01, -3.71624596e-01],
[-9.46347813e-01,  4.42308479e-01],
[ 6.54970714e-02,  1.53680672e-01],
[ 9.50046139e-01,  3.89984583e-01],
[ 5.58939215e-01,  8.09836879e-01],
[ 7.25701640e-01, -2.80462884e-01],
[ 2.03083726e+00,  1.20170801e-01],
[-1.51916239e-01,  9.58192690e-01],
[ 6.33253492e-01, -5.64187401e-01],
[ 2.02082877e+00,  5.18545302e-01],
[ 1.42594806e-02,  2.31861328e-01],
[-3.31332409e-01,  1.13198237e+00],
[ 9.10306386e-01,  2.01247355e-02],
[-3.81510263e-01,  9.71275515e-01],
[ 6.14424862e-01, -3.85865038e-01],
[ 1.98240620e+00,  3.50624890e-01],
[ 4.57707908e-01,  8.94353953e-01],
[ 1.03913732e+00,  3.89233822e-01],
[ 1.98947025e+00,  1.83673590e-01],
[-6.70964946e-01,  7.93803059e-01],
[ 2.01955021e+00,  3.25289242e-01],
[ 1.77991353e+00, -1.11649028e-01],
[ 1.74859902e+00, -2.65552069e-01],
[ 4.91433723e-01, -2.73866374e-01],
[-1.06817822e+00,  2.97496839e-01],
[ 5.66538453e-02,  2.68583098e-02],
[ 1.33536908e+00, -6.97952991e-01],
[ 1.98056273e+00,  3.68945922e-01],
[ 8.04337273e-02,  1.15888076e+00],
[ 1.06172312e+00, -4.77816364e-02],
[-8.13810949e-01,  4.55071382e-01],
```

```
[ 1.11442338e-01,  3.94660905e-01],
[-9.84200490e-01,  8.13605456e-02],
[-5.03822097e-01,  1.02687047e+00],
[ 1.56581318e+00, -3.50444504e-01],
[ 2.00175240e-01,  8.29137530e-01],
[-5.89576838e-01,  7.91803173e-01],
[ 5.91096314e-02,  1.04656810e+00],
[-8.64462243e-02,  1.14115330e+00],
[-8.03546811e-01,  3.68143826e-01],
[ 9.36602233e-01,  3.52975983e-01],
[ 1.50812484e+00, -5.68831767e-01],
[ 4.67190679e-01,  6.57821620e-01],
[ 1.97192261e+00,  1.74641987e-01],
[ 6.03754895e-01, -4.01384358e-01],
[ 6.81080119e-01,  4.27058398e-01],
[ 8.53498280e-01,  6.23877292e-01],
[-6.34861812e-01,  5.27478557e-01],
[-6.96765462e-02,  2.99484737e-01],
[ 1.95489528e-02,  9.94207504e-01],
[-1.26266935e-01,  3.43883486e-01],
[-8.39199835e-01,  7.00314293e-01],
[ 8.46496761e-01, -1.16350315e-02],
[ 2.29736915e-01,  7.69345906e-02],
[ 1.25462994e+00, -3.92336633e-01],
[ 1.96824632e+00,  5.49696837e-01],
[ 1.02016913e+00,  3.53917835e-01],
[-5.31164727e-01,  6.92582395e-01],
[-1.10496675e+00,  4.86301545e-01],
[ 2.54223306e-01, -1.89474183e-01],
[ 1.25509219e+00, -3.32432682e-01],
[ 1.57148223e+00, -8.26665774e-02],
[ 1.11207737e+00, -3.29369368e-01],
[-1.05050177e+00,  1.81123803e-01],
[ 2.03731913e+00,  5.06268525e-01],
[ 8.37104705e-01,  4.38321722e-01],
[ 2.70944977e-02,  4.29302511e-02],
[-2.68526403e-02,  5.04722011e-01],
[ 4.47263472e-01,  9.95926194e-01],
[-4.78109868e-01,  9.94572554e-01],
[-4.04629660e-01,  8.86029986e-01],
[ 1.03118702e+00,  1.02021446e-01],
[ 2.67391648e-01,  1.15977827e-01],
[ 1.31563032e+00, -4.68055845e-01],
[ 2.72709481e-01,  1.06929344e+00],
[-2.42902398e-02,  1.56393942e-01],
[ 4.77795869e-01, -3.90843212e-01],
[ 1.68292595e+00, -4.60914204e-01],
```

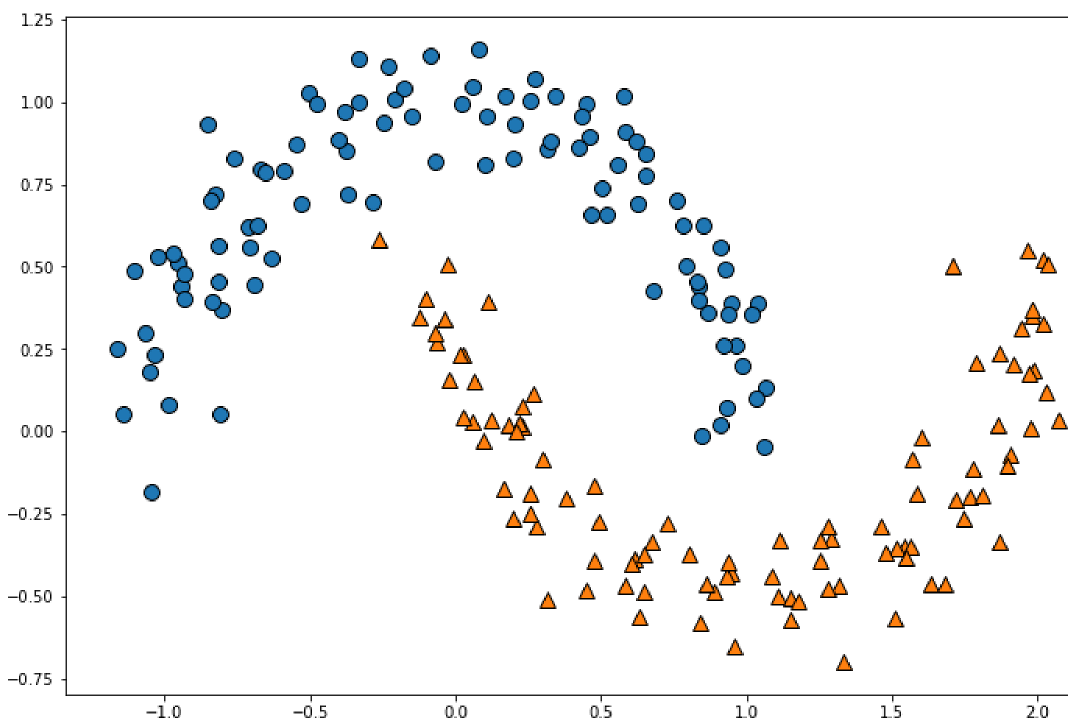
```
[ 8.33302236e-01,  3.98915558e-01],
[ 3.27488923e-01,  8.78131092e-01],
[ 6.74827352e-01, -3.35549427e-01],
[-2.45527648e-01,  9.38992348e-01],
[ 1.28155163e+00, -4.76190226e-01],
[ 3.01408950e-01, -8.43203922e-02],
[ 5.80344402e-01,  1.01606800e+00],
[ 6.54257532e-01,  7.78195644e-01],
[ 2.18332379e-01,  2.39832599e-02],
[ 1.17599534e+00, -5.14188256e-01],
[ 6.53498000e-01,  8.42628380e-01],
[ 1.46494235e+00, -2.88413814e-01],
[ 4.50729150e-01, -4.81782194e-01],
[ 1.59955423e+00, -1.97918553e-02],
[ 4.23800432e-01,  8.61219660e-01],
[ 6.47861265e-01, -3.73026171e-01],
[ 1.54832195e+00, -3.82636573e-01],
[ 9.59507036e-01, -6.53322764e-01],
[ 6.22396337e-01,  8.80914616e-01],
[-1.14279515e+00,  5.33863108e-02],
[-9.68196502e-01,  5.38851311e-01],
[-6.91219117e-01,  4.43566916e-01],
[ 1.82600270e-01,  1.76667308e-02],
[ 9.34822203e-01, -3.97721167e-01],
[ 1.70786789e+00,  5.01289734e-01],
[-6.81686209e-01,  6.25964062e-01],
[-7.06439154e-01,  5.56250067e-01],
[ 5.01454747e-01,  7.38339946e-01],
[ 1.47648298e+00, -3.69367080e-01],
[-9.33946572e-01,  4.76649526e-01],
[ 1.91815067e+00,  2.06055971e-01],
[ 1.87199356e+00, -3.36720051e-01],
[ 1.94311158e+00,  3.13223536e-01],
[ 3.42365360e-01,  1.01614096e+00],
[-7.12211574e-02,  8.20286056e-01],
[ 8.38200245e-01, -5.80104512e-01],
[-8.34317448e-01,  3.91907169e-01],
[-6.52120149e-01,  7.85982383e-01],
[-9.31767725e-01,  4.03435241e-01],
[ 4.33542139e-01,  9.55743482e-01],
[ 8.31174752e-01,  4.53839488e-01],
[ 9.20525579e-01,  2.60496062e-01],
[-3.67116967e-02,  3.42448285e-01],
[-5.45442229e-01,  8.70811316e-01],
[ 1.79020519e+00,  2.09022313e-01],
[ 2.09486633e-01, -7.84617937e-04],
[ 1.69222125e-01,  1.01894845e+00],
```

```
        [ 1.58819062e+00, -1.90820935e-01]]),
 array([0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1,
        0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1,
        1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0,
        0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0,
        1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0,
        0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0,
        0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0,
        1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1,
        1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1,
        0, 1]))
```

[19]:
```python
X = moons[0]
y = moons[1]
```

[20]:
```python
plt.figure(figsize=(12, 8))
mglearn.discrete_scatter(X[:, 0], X[:, 1], y)
```

[20]:
```
[<matplotlib.lines.Line2D at 0x7f441cb7e190>,
 <matplotlib.lines.Line2D at 0x7f441cba1f90>]
```

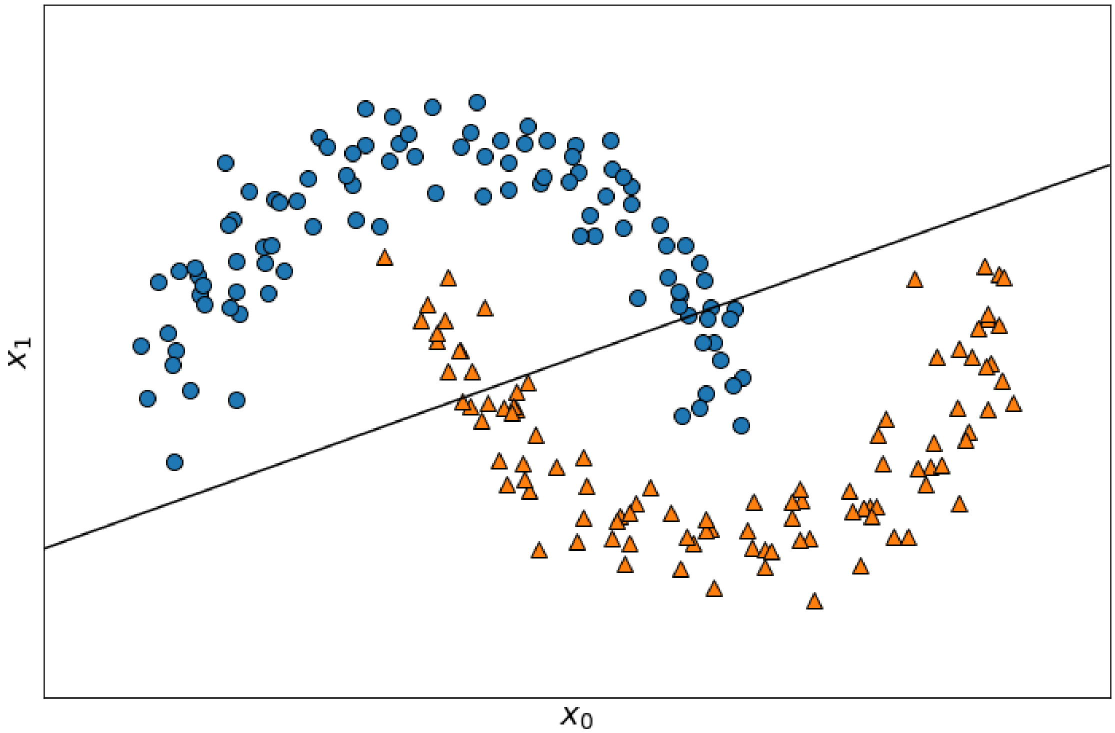


[21]:
```python
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import LinearSVC
```

[22]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y,␣
↪random_state=0)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.fit_transform(X_test)
```

[23]:
```
lin_svm = LinearSVC().fit(X_train_scaled, y_train)
```

[24]:
```
plt.figure(figsize=(12, 8))
mglearn.plots.plot_2d_separator(lin_svm, X)
mglearn.discrete_scatter(X[:, 0], X[:, 1], y)
plt.xlabel('$x_0$', fontsize=20)
plt.ylabel('$x_1$', fontsize=20)
plt.show()
```



[25]:
```
from sklearn.preprocessing import PolynomialFeatures
```

[26]:
```
poly = PolynomialFeatures(degree=3)
X_train_poly = poly.fit_transform(X_train)
X_test_poly = poly.fit_transform(X_test)

print(X_train_poly)
```

```python
print(X_test_poly)
```

```
[[ 1.00000000e+00  8.86813848e-01 -4.84896235e-01 … -3.81341214e-01
   2.08511538e-01 -1.14010917e-01]
 [ 1.00000000e+00  9.24799914e-01  4.94247065e-01 …  4.22707215e-01
   2.25910272e-01  1.20734753e-01]
 [ 1.00000000e+00 -7.63797985e-01  8.26528213e-01 …  4.82186114e-01
  -5.21787743e-01  5.64641828e-01]
 …
 [ 1.00000000e+00 -9.68196502e-01  5.38851311e-01 …  5.05121626e-01
  -2.81126249e-01  1.56461263e-01]
 [ 1.00000000e+00 -6.70964946e-01  7.93803059e-01 …  3.57365342e-01
  -4.22790644e-01  5.00193800e-01]
 [ 1.00000000e+00 -2.68526403e-02  5.04722011e-01 …  3.63937020e-04
  -6.84055729e-03  1.28575060e-01]]
[[ 1.00000000e+00  8.37104705e-01  4.38321722e-01  7.00744287e-01
   3.66921175e-01  1.92125932e-01  5.86596339e-01  3.07151442e-01
   1.60829521e-01  8.42129692e-02]
 [ 1.00000000e+00 -8.52697952e-01  9.32383274e-01  7.27093798e-01
  -7.95041309e-01  8.69338570e-01 -6.19991393e-01  6.77930096e-01
  -7.41283218e-01  8.10556742e-01]
 [ 1.00000000e+00  1.02016913e+00  3.53917835e-01  1.04074506e+00
   3.61056051e-01  1.25257834e-01  1.06173599e+00  3.68338239e-01
   1.27784176e-01  4.43309814e-02]
 [ 1.00000000e+00  4.47263472e-01  9.95926194e-01  2.00044614e-01
   4.45441408e-01  9.91868984e-01  8.94726486e-02  1.99229671e-01
   4.43626766e-01  9.87828302e-01]
 [ 1.00000000e+00  4.67190679e-01  6.57821620e-01  2.18267131e-01
   3.07328129e-01  4.32729284e-01  1.01972369e-01  1.43580837e-01
   2.02167088e-01  2.84658679e-01]
 [ 1.00000000e+00  3.79557530e-01 -2.02301374e-01  1.44063918e-01
  -7.67850098e-02  4.09258460e-02  5.46805450e-02 -2.91443287e-02
   1.55337130e-02 -8.27935487e-03]
 [ 1.00000000e+00  2.02082877e+00  5.18545302e-01  4.08374893e+00
   1.04789127e+00  2.68889230e-01  8.25255735e+00  2.11760882e+00
   5.43379093e-01  1.39431247e-01]
 [ 1.00000000e+00  7.83108552e-01  6.23681079e-01  6.13259005e-01
   4.88409987e-01  3.88978088e-01  4.80248371e-01  3.82478038e-01
   3.04612067e-01  2.42598273e-01]
 [ 1.00000000e+00  1.74859902e+00 -2.65552069e-01  3.05759853e+00
  -4.64344087e-01  7.05179012e-02  5.34651378e+00 -8.11951614e-01
   1.23307533e-01 -1.87261746e-02]
 [ 1.00000000e+00  1.89450511e+00 -1.02565542e-01  3.58914960e+00
  -1.94310943e-01  1.05196904e-02  6.79966225e+00 -3.68123074e-01
   1.99296071e-02 -1.07895774e-03]
 [ 1.00000000e+00  1.21605162e-01  3.53588146e-02  1.47878155e-02
   4.29981438e-03  1.25024577e-03  1.79827470e-03  5.22879625e-04
```

```
  1.52036340e-04  4.42072084e-05]
 [ 1.00000000e+00  1.90607398e+00 -7.10915927e-02  3.63311801e+00
  -1.35505835e-01  5.05401456e-03  6.92499170e+00 -2.58284146e-01
   9.63332563e-03 -3.59297944e-04]
 [ 1.00000000e+00 -1.10496675e+00  4.86301545e-01  1.22095151e+00
  -5.37347036e-01  2.36489193e-01 -1.34911082e+00  5.93750607e-01
  -2.61312694e-01  1.15005060e-01]
 [ 1.00000000e+00  6.74827352e-01 -3.35549427e-01  4.55391955e-01
  -2.26437931e-01  1.12593418e-01  3.07310947e-01 -1.52806509e-01
   7.59811179e-02 -3.77806568e-02]
 [ 1.00000000e+00  1.03579202e-01  8.07972963e-01  1.07286511e-02
   8.36891947e-02  6.52820309e-01  1.11126512e-03  8.66845999e-03
   6.76186066e-02  5.27461159e-01]
 [ 1.00000000e+00  1.87199356e+00 -3.36720051e-01  3.50435989e+00
  -6.30337767e-01  1.13380393e-01  6.56013914e+00 -1.17998824e+00
   2.12247365e-01 -3.81774517e-02]
 [ 1.00000000e+00 -2.45527648e-01  9.38992348e-01  6.02838258e-02
  -2.30548582e-01  8.81706629e-01 -1.48013459e-02  5.66060511e-02
  -2.16483355e-01  8.27915777e-01]
 [ 1.00000000e+00  6.47834107e-01 -4.86523627e-01  4.19689030e-01
  -3.15186599e-01  2.36705240e-01  2.71888868e-01 -2.04188629e-01
   1.53345728e-01 -1.15162692e-01]
 [ 1.00000000e+00  1.98240620e+00  3.50624890e-01  3.92993432e+00
   6.95080955e-01  1.22937814e-01  7.79072615e+00  1.37793279e+00
   2.43712683e-01  4.31050574e-02]
 [ 1.00000000e+00 -9.31767725e-01  4.03435241e-01  8.68191093e-01
  -3.75907937e-01  1.62759994e-01 -8.08952440e-01  3.50258883e-01
  -1.51654509e-01  6.56631174e-02]
 [ 1.00000000e+00  8.31174752e-01  4.53839488e-01  6.90851468e-01
   3.77219924e-01  2.05970281e-01  5.74218298e-01  3.13535677e-01
   1.71197297e-01  9.34774469e-02]
 [ 1.00000000e+00  1.57148223e+00 -8.26665774e-02  2.46955638e+00
  -1.29909057e-01  6.83376302e-03  3.88086396e+00 -2.04149774e-01
   1.07391371e-02 -5.64923799e-04]
 [ 1.00000000e+00  1.06619086e+00  1.33052899e-01  1.13676295e+00
   1.41859785e-01  1.77030740e-02  1.21200626e+00  1.51249606e-01
   1.88748557e-02  2.35544532e-03]
 [ 1.00000000e+00  1.97192261e+00  1.74641987e-01  3.88847879e+00
   3.44380483e-01  3.04998235e-02  7.66777927e+00  6.79091662e-01
   6.01432918e-02  5.32654978e-03]
 [ 1.00000000e+00 -4.04629660e-01  8.86029986e-01  1.63725162e-01
  -3.58514012e-01  7.85049136e-01 -6.62480568e-02  1.45065403e-01
  -3.17654165e-01  6.95577075e-01]
 [ 1.00000000e+00  2.29736915e-01  7.69345906e-02  5.27790499e-02
   1.76747155e-02  5.91893124e-03  1.21252961e-02  4.06053460e-03
   1.35979700e-03  4.55370552e-04]
 [ 1.00000000e+00  1.82600270e-01  1.76667308e-02  3.33428585e-02
   3.22594981e-03  3.12113376e-04  6.08841497e-03  5.89059305e-04
```

```
  5.69919867e-05  5.51402298e-06]
 [ 1.00000000e+00 -7.06439154e-01  5.56250067e-01  4.99056278e-01
  -3.92956826e-01  3.09414136e-01 -3.52552895e-01  2.77600088e-01
  -2.18582261e-01  1.72111634e-01]
 [ 1.00000000e+00  1.63158315e+00 -4.63896705e-01  2.66206358e+00
  -7.56886048e-01  2.15200153e-01  4.34337809e+00 -1.23492252e+00
   3.51116944e-01 -9.98306417e-02]
 [ 1.00000000e+00  8.46496761e-01 -1.16350315e-02  7.16556767e-01
  -9.84901649e-03  1.35373958e-04  6.06562982e-01 -8.33716056e-03
   1.14593617e-04 -1.57508027e-06]
 [ 1.00000000e+00  1.03913732e+00  3.89233822e-01  1.07980638e+00
   4.04467391e-01  1.51502968e-01  1.12206711e+00  4.20297162e-01
   1.57432388e-01  5.89700792e-02]
 [ 1.00000000e+00  1.97796688e+00  1.04796611e-02  3.91235297e+00
   2.07284225e-02  1.09823296e-04  7.73850460e+00  4.10001332e-02
   2.17226843e-04  1.15091092e-06]
 [ 1.00000000e+00  4.23800432e-01  8.61219660e-01  1.79606806e-01
   3.64985264e-01  7.41699304e-01  7.61174418e-02  1.54680912e-01
   3.14332485e-01  6.38766022e-01]
 [ 1.00000000e+00  6.03754895e-01 -4.01384358e-01  3.64519973e-01
  -2.42337771e-01  1.61109403e-01  2.20080718e-01 -1.46312615e-01
   9.72705907e-02 -6.46667944e-02]
 [ 1.00000000e+00 -1.06817822e+00  2.97496839e-01  1.14100470e+00
  -3.17779643e-01  8.85043693e-02 -1.21879637e+00  3.39445292e-01
  -9.45384393e-02  2.63297701e-02]
 [ 1.00000000e+00  2.01955021e+00  3.25289242e-01  4.07858306e+00
   6.56937957e-01  1.05813091e-01  8.23690328e+00  1.32671919e+00
   2.13694850e-01  3.44198601e-02]
 [ 1.00000000e+00  5.01454747e-01  7.38339946e-01  2.51456863e-01
   3.70244071e-01  5.45145877e-01  1.26094237e-01  1.85660647e-01
   2.73365987e-01  4.02502977e-01]
 [ 1.00000000e+00 -1.51916239e-01  9.58192690e-01  2.30785436e-02
  -1.45565029e-01  9.18133230e-01 -3.50600554e-03  2.21136918e-02
  -1.39479347e-01  8.79748549e-01]
 [ 1.00000000e+00  1.76704822e+00 -1.98609868e-01  3.12245941e+00
  -3.50953214e-01  3.94458798e-02  5.51753634e+00 -6.20151253e-01
   6.97027717e-02 -7.83434100e-03]
 [ 1.00000000e+00  2.03083726e+00  1.20170801e-01  4.12429999e+00
   2.44047341e-01  1.44410215e-02  8.37578210e+00  4.95620434e-01
   2.93273645e-02  1.73538912e-03]
 [ 1.00000000e+00 -9.46347813e-01  4.42308479e-01  8.95574184e-01
  -4.18577662e-01  1.95636791e-01 -8.47524671e-01  3.96120055e-01
  -1.85140449e-01  8.65318113e-02]
 [ 1.00000000e+00  1.51596993e+00 -3.52163413e-01  2.29816484e+00
  -5.33869146e-01  1.24019069e-01  3.48394880e+00 -8.09329574e-01
   1.88009180e-01 -4.36749787e-02]
 [ 1.00000000e+00  1.31563032e+00 -4.68055845e-01  1.73088315e+00
  -6.15788462e-01  2.19076274e-01  2.27720235e+00 -8.10149973e-01
```

```
  2.88223389e-01 -1.02539930e-01]
 [ 1.00000000e+00  9.50046139e-01  3.89984583e-01  9.02587666e-01
   3.70503347e-01  1.52087975e-01  8.57499927e-01  3.51995274e-01
   1.44490593e-01  5.93119654e-02]
 [ 1.00000000e+00 -3.30952410e-01  1.00049623e+00  1.09529498e-01
  -3.31116640e-01  1.00099272e+00 -3.62490512e-02  1.09583850e-01
  -3.31280951e-01  1.00148944e+00]
 [ 1.00000000e+00  8.04337273e-02  1.15888076e+00  6.46958449e-03
   9.32130992e-02  1.34300462e+00  5.20372794e-04  7.49747701e-03
   1.08022868e-01  1.55638222e+00]
 [ 1.00000000e+00  6.33253492e-01 -5.64187401e-01  4.01009985e-01
  -3.57273641e-01  3.18307423e-01  2.53940973e-01 -2.26244781e-01
   2.01569287e-01 -1.79585038e-01]
 [ 1.00000000e+00 -1.14279515e+00  5.33863108e-02  1.30598075e+00
  -6.10096170e-02  2.85009819e-03 -1.49246846e+00  6.97214942e-02
  -3.25707838e-03  1.52156228e-04]
 [ 1.00000000e+00 -1.00520615e-01  4.03688699e-01  1.01043941e-02
  -4.05790365e-02  1.62964566e-01 -1.01569992e-03  4.07902972e-03
  -1.63812984e-02  6.57869535e-02]
 [ 1.00000000e+00  4.77778324e-01 -1.64382571e-01  2.28272127e-01
  -7.85384295e-02  2.70216298e-02  1.09063474e-01 -3.75239593e-02
   1.29103490e-02 -4.44188499e-03]]
```

[27]:
```
poly.get_feature_names()
```

[27]:
```
['1',
 'x0',
 'x1',
 'x0^2',
 'x0 x1',
 'x1^2',
 'x0^3',
 'x0^2 x1',
 'x0 x1^2',
 'x1^3']
```

[28]:
```
X_train_poly_scaled = scaler.fit_transform(X_train_poly)
X_test_poly_scaled = scaler.fit_transform(X_test_poly)

lin_svm = LinearSVC().fit(X_train_poly_scaled, y_train)
lin_svm.predict(X_test_poly_scaled) == y_test
```

[28]:
```
array([ True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
```

```
        True,  True,  True,  True,  True])
```

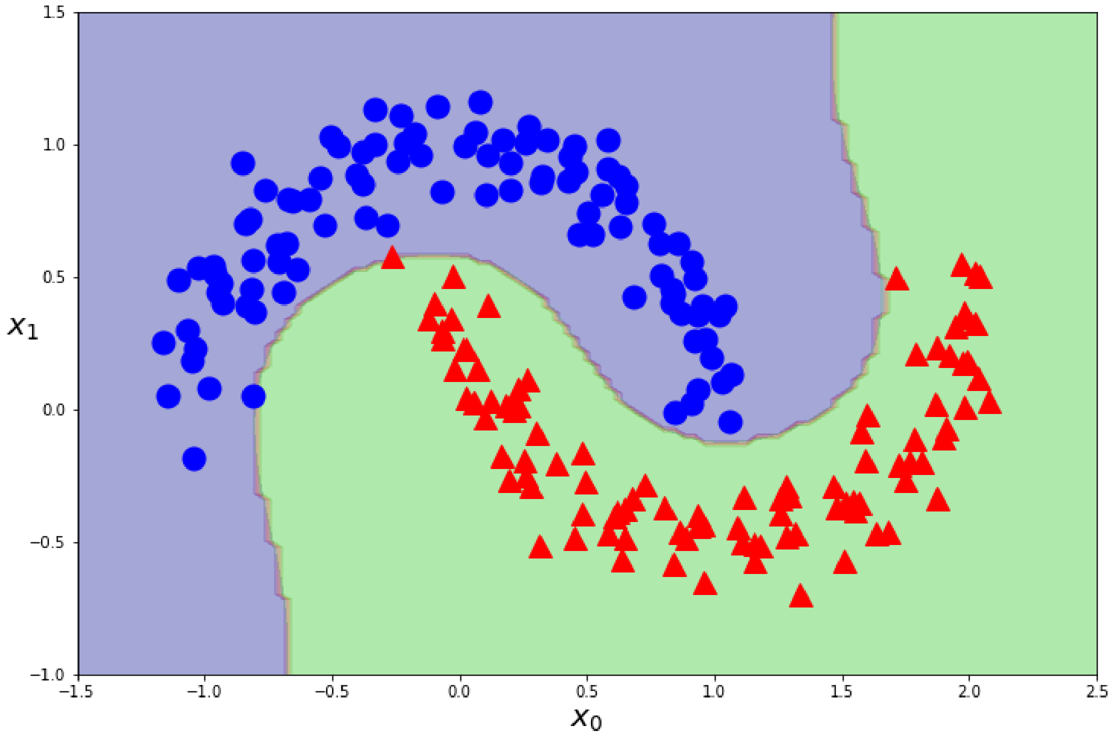
[29]:
```
from sklearn.pipeline import Pipeline

poly_svm = Pipeline([
    ('poly', PolynomialFeatures(degree=3)),
    ('scaler', StandardScaler()),
    ('svm', LinearSVC())
])

poly_svm.fit(X, y)
```

[29]:
```
Pipeline(memory=None,
         steps=[('poly',
                 PolynomialFeatures(degree=3, include_bias=True,
                                    interaction_only=False, order='C')),
                ('scaler',
                 StandardScaler(copy=True, with_mean=True, with_std=True)),
                ('svm',
                 LinearSVC(C=1.0, class_weight=None, dual=True,
                           fit_intercept=True, intercept_scaling=1,
                           loss='squared_hinge', max_iter=1000,
                           multi_class='ovr', penalty='l2', random_state=None,
                           tol=0.0001, verbose=0))],
         verbose=False)
```

[30]:
```
def plot_decision_function(model):
    _x0 = np.linspace(-1.5, 2.5, 100)
    _x1 = np.linspace(-1.0, 1.5, 100)
    x0, x1 = np.meshgrid(_x0, _x1)
    X = np.c_[x0.ravel(), x1.ravel()]
    y_pred = model.predict(X).reshape(x0.shape)
    y_decision = model.decision_function(X).reshape(x0.shape)
    plt.contourf(x0, x1, y_pred, cmap=plt.cm.brg, alpha=0.2)
    plt.contourf(x0, x1, y_decision, levels=[y_decision.min(), 0, y_decision.
 ↪max()], alpha=0.3)
```

[31]:
```
def plot_dataset(X, y):
    plt.plot(X[:, 0][y==0], X[:, 1][y==0], 'bo', ms=15)
    plt.plot(X[:, 0][y==1], X[:, 1][y==1], 'r^', ms=15)
    plt.xlabel('$x_0$', fontsize=20)
    plt.ylabel('$x_1$', fontsize=20, rotation=0)
```

[32]:
```
plt.figure(figsize=(12, 8))
plot_decision_function(poly_svm)
plot_dataset(X, y)
plt.show()
```



[33]:
```
from sklearn.svm import SVC
```

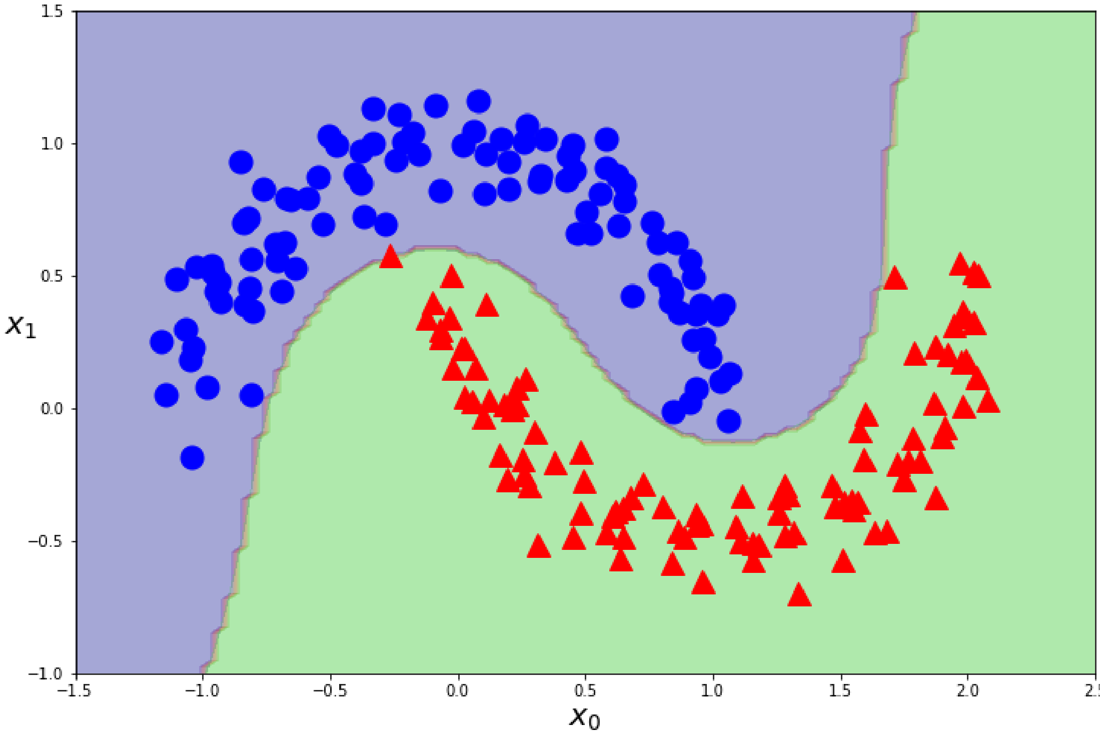
[34]:
```
kernel_svm = Pipeline([
    ('scaler', StandardScaler()),
    ('svm', SVC(kernel='poly', degree=3, coef0=1))
])
```

[35]:
```
kernel_svm.fit(X, y)
```

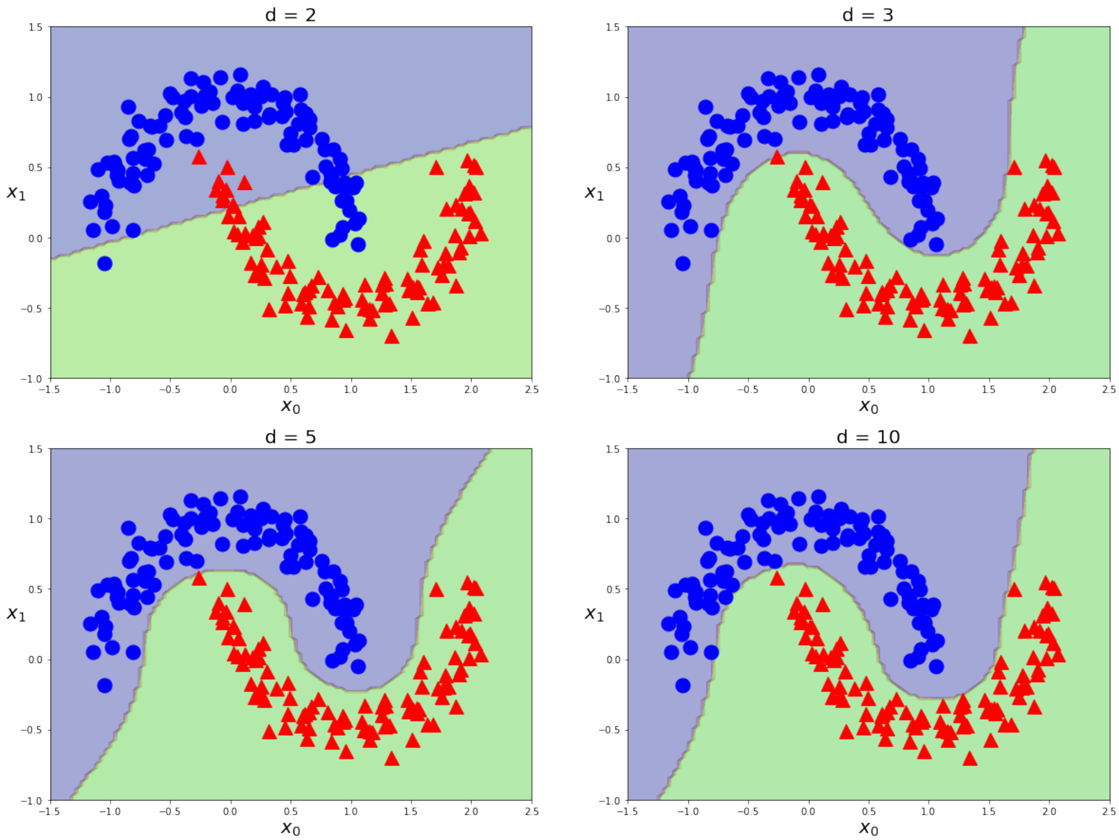
[35]:
```
Pipeline(memory=None,
         steps=[('scaler',
                 StandardScaler(copy=True, with_mean=True, with_std=True)),
                ('svm',
                 SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None,
                     coef0=1, decision_function_shape='ovr', degree=3,
                     gamma='scale', kernel='poly', max_iter=-1,
                     probability=False, random_state=None, shrinking=True,
                     tol=0.001, verbose=False))],
         verbose=False)
```

[36]:
```
plt.figure(figsize=(12, 8))
plot_decision_function(kernel_svm)
plot_dataset(X, y)
```

```python
plt.show()
```



[37]:
```python
plt.figure(figsize=(20, 15))

for i, degree in enumerate([2, 3, 5, 10]):
    poly_kernel_svm = Pipeline([
        ('scaler', StandardScaler()),
        ('svm', SVC(kernel='poly', degree=degree, coef0=1))
    ])
    poly_kernel_svm.fit(X, y)

    plt.subplot(221 + i)
    plot_decision_function(poly_kernel_svm)
    plot_dataset(X, y)
    plt.title('d = {}'.format(degree), fontsize=20)

plt.show()
```


d = 2
d = 3
d = 5
d = 10
x_1
x_0