In [6]:

```
import numpy as np
import sklearn
import matplotlib.pyplot as plt
import pandas as pd
import mglearn
```

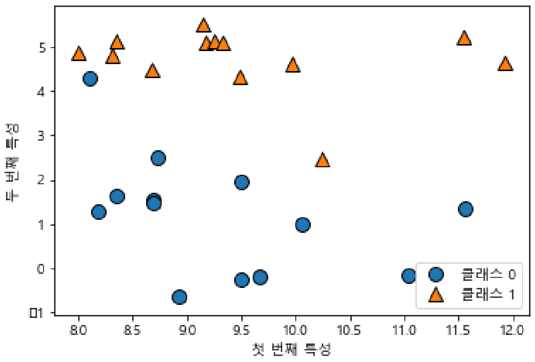
In [23]:

```
from matplotlib import font_manager, rc
font_name = font_manager.FontProperties(fname="c:/Windows/Fonts/malgun.ttf").get_name()
rc('font', family=font_name)
```

In [24]:

```
X,y = mglearn.datasets.make_forge()
mglearn.discrete_scatter(X[:, 0], X[:, 1], y)
plt.legend(["클래스 0", "클래스 1"], loc=4)
plt.xlabel("첫 번째 특성")
plt.ylabel("두 번째 특성")
print("X.shape:", X.shape)
```

```
X.shape: (26, 2)

C:₩ProgramData₩Anaconda3₩lib₩site-packages₩sklearn₩utils₩deprecation.py:77: Deprec
ationWarning: Function make_blobs is deprecated; Please import make_blobs directly
from scikit-learn
  warnings.warn(msg, category=DeprecationWarning)
```

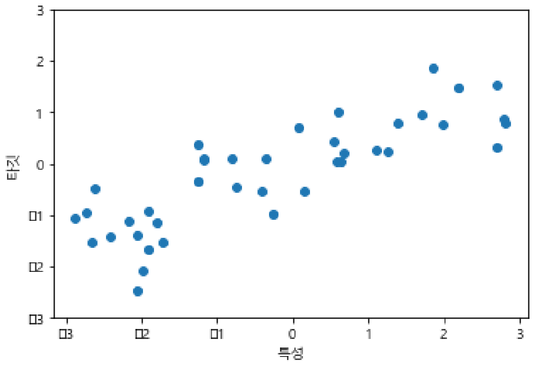


In [25]:

```
X, y = mglearn.datasets.make_wave(n_samples=40)
plt.plot(X, y, 'o')
plt.ylim(-3, 3)
plt.xlabel("특성")
plt.ylabel("타깃")
```

Out[25]:

```
Text(0, 0.5, '타깃')
```



In [27]:

```
from sklearn.datasets import load_breast_cancer
cancer = load_breast_cancer()
print("cancer.keys():₩n",cancer.keys())
```

```
cancer.keys():
 dict_keys(['data', 'target', 'target_names', 'DESCR', 'feature_names', 'filenam
e'])
```

In [28]:

```
print("유방암 데이터의 형태:", cancer.data.shape)
```

```
유방암 데이터의 형태: (569, 30)
```

In [30]:

```
print("클래스별 샘플 개수:₩n", {n: v for n, v in zip(cancer.target_names, np.bincount(cancer.ta
rget))})
```

```
클래스별 샘플 개수:
 {'malignant': 212, 'benign': 357}
```

In [31]:

```
print("특성 이름:₩n", cancer.feature_names)
```

```
특성 이름:
 ['mean radius' 'mean texture' 'mean perimeter' 'mean area'
 'mean smoothness' 'mean compactness' 'mean concavity'
 'mean concave points' 'mean symmetry' 'mean fractal dimension'
 'radius error' 'texture error' 'perimeter error' 'area error'
 'smoothness error' 'compactness error' 'concavity error'
 'concave points error' 'symmetry error' 'fractal dimension error'
 'worst radius' 'worst texture' 'worst perimeter' 'worst area'
 'worst smoothness' 'worst compactness' 'worst concavity'
 'worst concave points' 'worst symmetry' 'worst fractal dimension']
```

In [32]:

```
from sklearn.datasets import load_boston

boston = load_boston()
print("데이터의 형태:", boston.data.shape)
```

```
데이터의 형태: (506, 13)
```

In [33]:

```
X, y = mglearn.datasets.load_extended_boston()
print("X.shape:", X.shape)
```

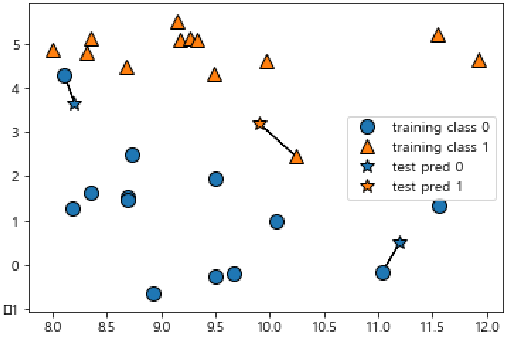
```
X.shape: (506, 104)
```

K-최근접 이웃

In [34]:

```
mglearn.plots.plot_knn_classification(n_neighbors=1)
```

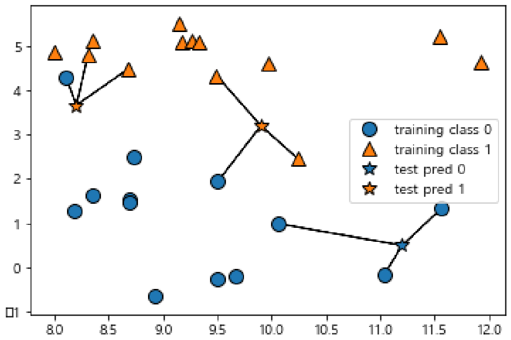
```
C:₩ProgramData₩Anaconda3₩lib₩site-packages₩sklearn₩utils₩deprecation.py:77: Deprec
ationWarning: Function make_blobs is deprecated; Please import make_blobs directly
from scikit-learn
  warnings.warn(msg, category=DeprecationWarning)
```



In [35]:

```
mglearn.plots.plot_knn_classification(n_neighbors=3)
```

```
C:₩ProgramData₩Anaconda3₩lib₩site-packages₩sklearn₩utils₩deprecation.py:77: Deprec
ationWarning: Function make_blobs is deprecated; Please import make_blobs directly
from scikit-learn
  warnings.warn(msg, category=DeprecationWarning)
```



In [36]:

```
from sklearn.model_selection import train_test_split
X, y = mglearn.datasets.make_forge()

X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)
```

```
C:₩ProgramData₩Anaconda3₩lib₩site-packages₩sklearn₩utils₩deprecation.py:77: Deprec
ationWarning: Function make_blobs is deprecated; Please import make_blobs directly
from scikit-learn
  warnings.warn(msg, category=DeprecationWarning)
```

In [37]:

```
from sklearn.neighbors import KNeighborsClassifier

clf = KNeighborsClassifier(n_neighbors=3)
```

In [38]:

```
clf.fit(X_train, y_train)
```

Out[38]:

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=None, n_neighbors=3, p=2,
           weights='uniform')
```

In [39]:

```
print("테스트 세트 예측:", clf.predict(X_test))
```

```
테스트 세트 예측: [1 0 1 0 1 0 0]
```

In [41]:

```
print("테스트 세트 정확도: {:.2f}".format(clf.score(X_test, y_test)))
```

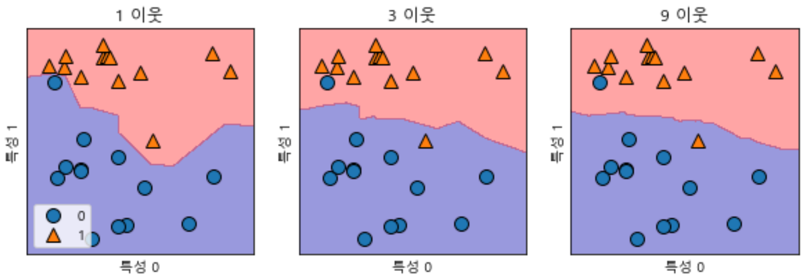
```
테스트 세트 정확도: 0.86
```

In [44]:

```
fig, axes = plt.subplots(1, 3, figsize=(10,3))

for n_neighbors, ax in zip([1, 3, 9], axes):
    clf = KNeighborsClassifier(n_neighbors=n_neighbors).fit(X, y)
    mglearn.plots.plot_2d_separator(clf, X, fill=True, eps=0.5, ax=ax, alpha=.4)
    mglearn.discrete_scatter(X[:, 0], X[:, 1], y, ax=ax)
    ax.set_title("{} 이웃".format(n_neighbors))
    ax.set_xlabel("특성 0")
    ax.set_ylabel("특성 1")
axes[0].legend(loc=3)
```

Out[44]:

```
<matplotlib.legend.Legend at 0x1ebb1a85ba8>
```

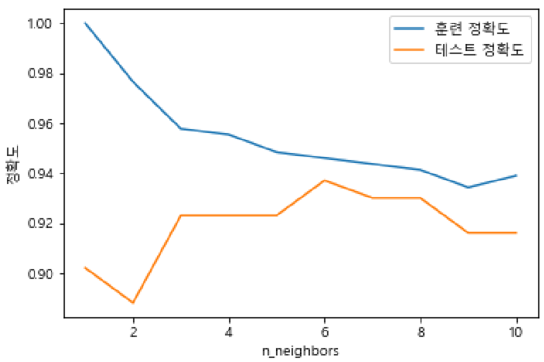


In [45]:

```
from sklearn.datasets import load_breast_cancer

cancer = load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(
    cancer.data, cancer.target, stratify=cancer.target, random_state=66)

training_accuracy = []
test_accuracy = []
neighbors_settings = range(1, 11)

for n_neighbors in neighbors_settings:
    clf = KNeighborsClassifier(n_neighbors=n_neighbors)
    clf.fit(X_train, y_train)
    training_accuracy.append(clf.score(X_train, y_train))
    test_accuracy.append(clf.score(X_test, y_test))

plt.plot(neighbors_settings, training_accuracy, label="훈련 정확도")
plt.plot(neighbors_settings, test_accuracy, label="테스트 정확도")
plt.ylabel("정확도")
plt.xlabel("n_neighbors")
plt.legend()
```

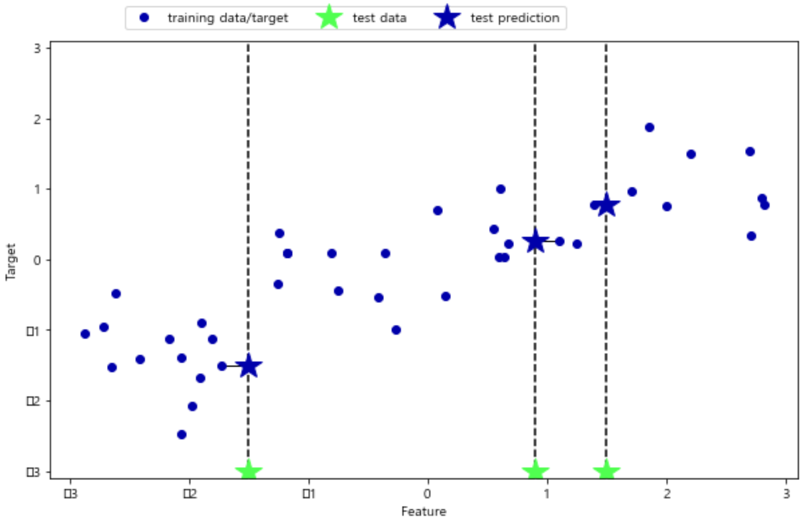
Out[45]:

```
<matplotlib.legend.Legend at 0x1ebb11ec9b0>
```



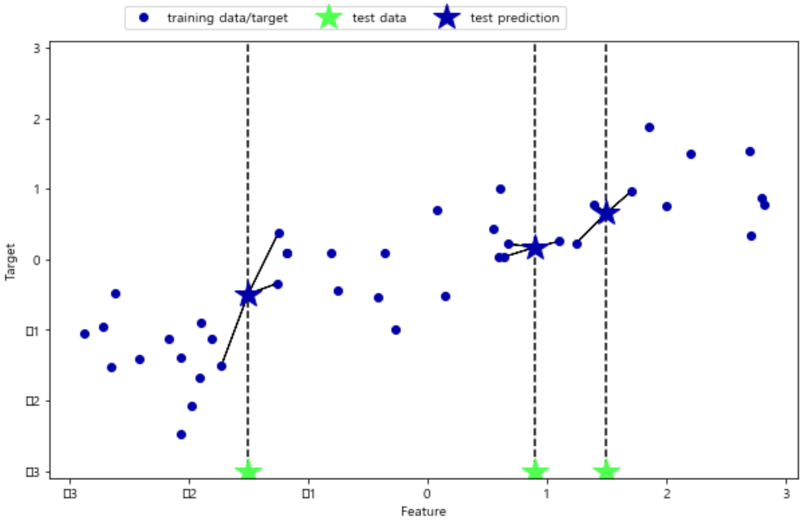
In [46]:

```
mglearn.plots.plot_knn_regression(n_neighbors=1)
```



In [47]:

```
mglearn.plots.plot_knn_regression(n_neighbors=3)
```



In [48]:

```
from sklearn.neighbors import KNeighborsRegressor

X, y = mglearn.datasets.make_wave(n_samples=40)
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)
reg = KNeighborsRegressor(n_neighbors=3)
reg.fit(X_train, y_train)
```

Out[48]:

```
KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski',
          metric_params=None, n_jobs=None, n_neighbors=3, p=2,
          weights='uniform')
```

In [58]:

```
print("테스트 세트 예측:₩n", reg.predict(X_test))
```

```
테스트 세트 예측:
 [-0.05396539  0.35686046  1.13671923 -1.89415682 -1.13881398 -1.63113382
  0.35686046  0.91241374 -0.44680446 -1.13881398]
```

In [59]:

```
print("테스트 세트 R^2: {:.2f}".format(reg.score(X_test, y_test)))
```

```
테스트 세트 R^2: 0.83
```

In [ ]:

```

```