Chapter_7_Combining_Different_Models_for_Ensemble_Learning

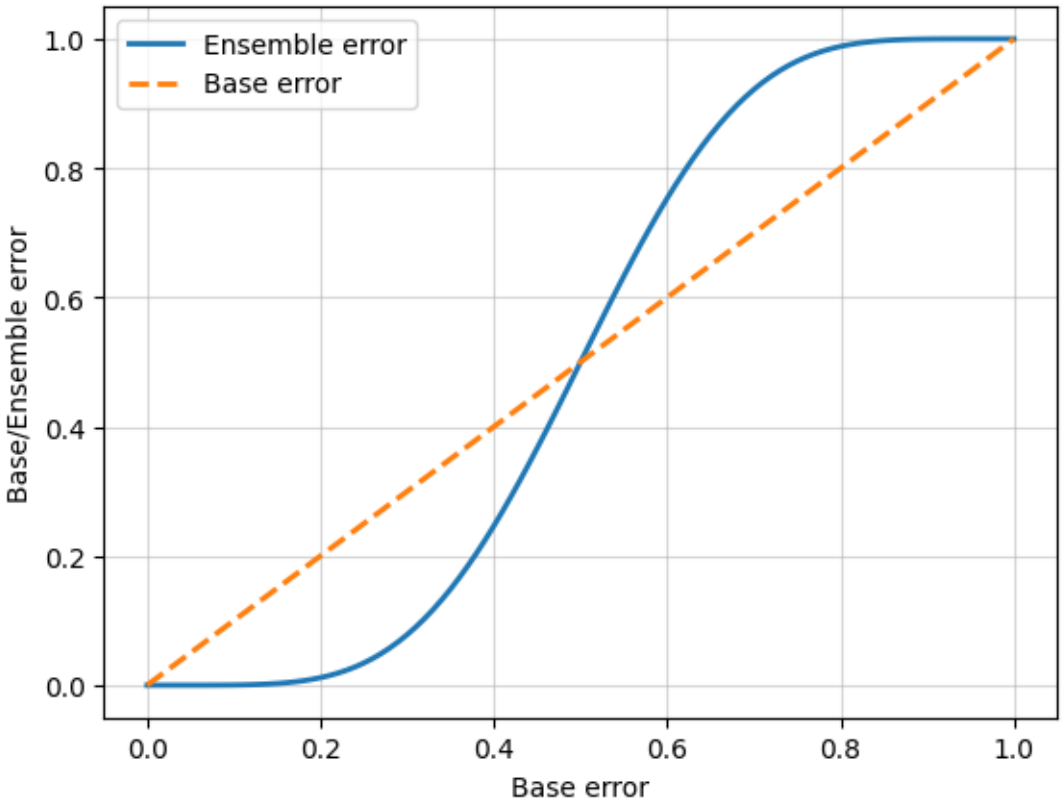
March 19, 2024

0.1 Learning with ensembles

[1]:
```
#implement the probability mass function in Python:
from scipy.special import comb
import math
def ensemble_error(n_classifier, error):
    k_start = int(math.ceil(n_classifier / 2.))
    probs = [comb(n_classifier, k) * error**k *(1-error)**(n_classifier - k)␣
 ↪for k in range(k_start, n_classifier + 1)]
    return sum(probs)
ensemble_error(n_classifier=11, error=0.25)
```

[1]:
```
0.03432750701904297
```

[2]:
```
import numpy as np
import matplotlib.pyplot as plt
error_range = np.arange(0.0, 1.01, 0.01)
ens_errors = [ensemble_error(n_classifier=11, error=error) for error in␣
 ↪error_range]
plt.plot(error_range, ens_errors,label='Ensemble error',linewidth=2)
plt.plot(error_range, error_range,linestyle='--', label='Base␣
 ↪error',linewidth=2)
plt.xlabel('Base error')
plt.ylabel('Base/Ensemble error')
plt.legend(loc='upper left')
plt.grid(alpha=0.5)
plt.show()
```



0.2 Implementing a Sinple Majority Vote Classifier

[3]:
```
ex = np.array([[0.9, 0.1],[0.8, 0.2],[0.4, 0.6]])
p = np.average(ex, axis=0, weights=[0.2, 0.2, 0.6])
print(p)
print(np.argmax(p))
```

```
[0.58 0.42]
0
```

[7]:
```
from sklearn.base import BaseEstimator
from sklearn.base import ClassifierMixin
from sklearn.preprocessing import LabelEncoder
import six
from sklearn.base import clone
from sklearn.pipeline import _name_estimators
import numpy as np
import operator
class MajorityVoteClassifier(BaseEstimator,ClassifierMixin):
    """ A majority vote ensemble classifier
    Parameters
```

```
    ----------
    classifiers : array-like, shape = [n_classifiers]
    Different classifiers for the ensemble
    vote : str, {'classlabel', 'probability'}
    Default: 'classlabel'
    If 'classlabel' the prediction is based on
    the argmax of class labels. Else if
    'probability', the argmax of the sum of
    probabilities is used to predict the class label
    (recommended for calibrated classifiers).
    weights : array-like, shape = [n_classifiers]
    Optional, default: None
    If a list of `int` or `float` values are
    provided, the classifiers are weighted by
    importance; Uses uniform weights if `weights=None`.
    """
    def __init__(self, classifiers,vote='classlabel', weights=None):
        self.classifiers = classifiers
        self.named_classifiers = {key: value for key, value in␣
↪_name_estimators(classifiers)}
        self.vote = vote
        self.weights = weights
    def fit(self, X, y):
        """ Fit classifiers.
        Parameters
        ----------
        X : {array-like, sparse matrix},
        shape = [n_samples, n_features]
        Matrix of training samples.
        y : array-like, shape = [n_samples]
        Vector of target class labels.
        Returns
        -------
        self : object
        """
        # Use LabelEncoder to ensure class labels start
        # with 0, which is important for np.argmax
        # call in self.predict
        self.lablenc_ = LabelEncoder()
        self.lablenc_.fit(y)
        self.classes_ = self.lablenc_.classes_
        self.classifiers_ = []
        for clf in self.classifiers:
            fitted_clf = clone(clf).fit(X,self.lablenc_.transform(y))
            self.classifiers_.append(fitted_clf)
        return self
```

[8]:
```
def predict(self, X):
    """ Predict class labels for X.
    Parameters
    ----------
    X : {array-like, sparse matrix},
    Shape = [n_samples, n_features]
    Matrix of training samples.
    Returns
    ----------
    maj_vote : array-like, shape = [n_samples]
    Predicted class labels.
    """
    if self.vote == 'probability':
        maj_vote = np.argmax(self.predict_proba(X),axis=1)
    else: # 'classlabel' vote
        # Collect results from clf.predict calls
        predictions = np.asarray([clf.predict(X) for clf in self.classifiers_]).
 ↪T
        maj_vote = np.apply_along_axis(lambda x: np.argmax(np.bincount(x,␣
 ↪weights=self.weights)), axis=1,arr=predictions)
    maj_vote = self.lablenc_.inverse_transform(maj_vote)
    return maj_vote
def predict_proba(self, X):
    """ Predict class probabilities for X.
    Parameters
    ----------
    X : {array-like, sparse matrix},
    shape = [n_samples, n_features]
    Training vectors, where n_samples is
    the number of samples and
    n_features is the number of features.
    Returns
    ----------
    avg_proba : array-like,
    shape = [n_samples, n_classes]
    Weighted average probability for
    each class per sample.
    """
    probas = np.asarray([clf.predict_proba(X) for clf in self.classifiers_])
    avg_proba = np.average(probas, axis=0, weights=self.weights)
    return avg_proba
def get_params(self, deep=True):
    """ Get classifier parameter names for GridSearch"""
    if not deep:
        return super(MajorityVoteClassifier,self).get_params(deep=False)
    else:
        out = self.named_classifiers.copy()
```

```
        for name, step in six.iteritems(self.named_classifiers):
            for key, value in six.iteritems(step.get_params(deep=True)):
                out['%s__%s' % (name, key)] = value
        return out
```

0.3 Using the majority voting principle to make predictions

[9]:
```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
iris = datasets.load_iris()
X, y = iris.data[50:, [1, 2]], iris.target[50:]
le = LabelEncoder()
y = le.fit_transform(y)
```

[10]:
```
X_train, X_test, y_train, y_test =train_test_split(X, y,test_size=0.
 ↪5,random_state=1,stratify=y)
```

[11]:
```
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import Pipeline
import numpy as np
clf1 = LogisticRegression(penalty='l2',C=0.001,random_state=1)
clf2 = DecisionTreeClassifier(max_depth=1,criterion='entropy',random_state=0)
clf3 = KNeighborsClassifier(n_neighbors=1,p=2,metric='minkowski')
pipe1 = Pipeline([['sc', StandardScaler()],['clf', clf1]])
pipe3 = Pipeline([['sc', StandardScaler()],['clf', clf3]])
clf_labels = ['Logistic regression', 'Decision tree', 'KNN']
print('10-fold cross validation:\n')
for clf, label in zip([pipe1, clf2, pipe3], clf_labels):
    scores =␣
 ↪cross_val_score(estimator=clf,X=X_train,y=y_train,cv=10,scoring='roc_auc')
    print("ROC AUC: %0.2f (+/- %0.2f) [%s]"% (scores.mean(), scores.std(),␣
 ↪label))
```

```
10-fold cross validation:

ROC AUC: 0.92 (+/- 0.15) [Logistic regression]
ROC AUC: 0.87 (+/- 0.18) [Decision tree]
ROC AUC: 0.85 (+/- 0.13) [KNN]
```

[12]:
```
mv_clf = MajorityVoteClassifier(classifiers=[pipe1, clf2, pipe3])
clf_labels += ['Majority voting']
all_clf = [pipe1, clf2, pipe3, mv_clf]
```

```
for clf, label in zip(all_clf, clf_labels):
    scores =␣
 ↪cross_val_score(estimator=clf,X=X_train,y=y_train,cv=10,scoring='roc_auc')
    print("Accuracy: %0.2f (+/- %0.2f) [%s]"% (scores.mean(), scores.std(),␣
 ↪label))
```

```
Accuracy: 0.92 (+/- 0.15) [Logistic regression]
Accuracy: 0.87 (+/- 0.18) [Decision tree]
Accuracy: 0.85 (+/- 0.13) [KNN]
Accuracy: nan (+/- nan) [Majority voting]
```

```
C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Desktop\Sel
f_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py:700: UserWarning: Scoring
failed. The score on this train-test partition for these parameters will be set
to nan. Details:
Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 334, in _score
    y_pred = method_caller(clf, "decision_function", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute
'decision_function'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py", line 687, in _score
    scores = scorer(estimator, X_test, y_test)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 88, in __call__
    *args, **kwargs)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 350, in _score
    y_pred = method_caller(clf, "predict_proba", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute 'predict_proba'
```

```
  UserWarning,
C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Desktop\Sel
f_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py:700: UserWarning: Scoring
failed. The score on this train-test partition for these parameters will be set
to nan. Details:
Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 334, in _score
    y_pred = method_caller(clf, "decision_function", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute
'decision_function'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py", line 687, in _score
    scores = scorer(estimator, X_test, y_test)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 88, in __call__
    *args, **kwargs)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 350, in _score
    y_pred = method_caller(clf, "predict_proba", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute 'predict_proba'

  UserWarning,
C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Desktop\Sel
f_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py:700: UserWarning: Scoring
failed. The score on this train-test partition for these parameters will be set
to nan. Details:
Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
```

```
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 334, in _score
    y_pred = method_caller(clf, "decision_function", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute
'decision_function'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py", line 687, in _score
    scores = scorer(estimator, X_test, y_test)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 88, in __call__
    *args, **kwargs)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 350, in _score
    y_pred = method_caller(clf, "predict_proba", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute 'predict_proba'

  UserWarning,
C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Desktop\Sel
f_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py:700: UserWarning: Scoring
failed. The score on this train-test partition for these parameters will be set
to nan. Details:
Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 334, in _score
    y_pred = method_caller(clf, "decision_function", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute
'decision_function'
```

```
During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py", line 687, in _score
    scores = scorer(estimator, X_test, y_test)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 88, in __call__
    *args, **kwargs)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 350, in _score
    y_pred = method_caller(clf, "predict_proba", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute 'predict_proba'

  UserWarning,
C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Desktop\Sel
f_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py:700: UserWarning: Scoring
failed. The score on this train-test partition for these parameters will be set
to nan. Details:
Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 334, in _score
    y_pred = method_caller(clf, "decision_function", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute
'decision_function'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py", line 687, in _score
    scores = scorer(estimator, X_test, y_test)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
```

```
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 88, in __call__
    *args, **kwargs)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 350, in _score
    y_pred = method_caller(clf, "predict_proba", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute 'predict_proba'

  UserWarning,
C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Desktop\Sel
f_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py:700: UserWarning: Scoring
failed. The score on this train-test partition for these parameters will be set
to nan. Details:
Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 334, in _score
    y_pred = method_caller(clf, "decision_function", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute
'decision_function'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py", line 687, in _score
    scores = scorer(estimator, X_test, y_test)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 88, in __call__
    *args, **kwargs)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 350, in _score
    y_pred = method_caller(clf, "predict_proba", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
```

```
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute 'predict_proba'

  UserWarning,
C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Desktop\Sel
f_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py:700: UserWarning: Scoring
failed. The score on this train-test partition for these parameters will be set
to nan. Details:
Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 334, in _score
    y_pred = method_caller(clf, "decision_function", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute
'decision_function'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py", line 687, in _score
    scores = scorer(estimator, X_test, y_test)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 88, in __call__
    *args, **kwargs)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 350, in _score
    y_pred = method_caller(clf, "predict_proba", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute 'predict_proba'

  UserWarning,
C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Desktop\Sel
f_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py:700: UserWarning: Scoring
failed. The score on this train-test partition for these parameters will be set
```

```
to nan. Details:
Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 334, in _score
    y_pred = method_caller(clf, "decision_function", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute
'decision_function'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py", line 687, in _score
    scores = scorer(estimator, X_test, y_test)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 88, in __call__
    *args, **kwargs)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 350, in _score
    y_pred = method_caller(clf, "predict_proba", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute 'predict_proba'

  UserWarning,
C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Desktop\Sel
f_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py:700: UserWarning: Scoring
failed. The score on this train-test partition for these parameters will be set
to nan. Details:
Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 334, in _score
    y_pred = method_caller(clf, "decision_function", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
```

```
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute
'decision_function'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py", line 687, in _score
    scores = scorer(estimator, X_test, y_test)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 88, in __call__
    *args, **kwargs)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 350, in _score
    y_pred = method_caller(clf, "predict_proba", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute 'predict_proba'

  UserWarning,
C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Desktop\Sel
f_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\model_selection\_validation.py:700: UserWarning: Scoring
failed. The score on this train-test partition for these parameters will be set
to nan. Details:
Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 334, in _score
    y_pred = method_caller(clf, "decision_function", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute
'decision_function'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
```

```
packages\sklearn\model_selection\_validation.py", line 687, in _score
    scores = scorer(estimator, X_test, y_test)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 88, in __call__
    *args, **kwargs)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 350, in _score
    y_pred = method_caller(clf, "predict_proba", X)
  File "C:\Users\ankit19.gupta\OneDrive - Reliance Corporate IT Park Limited\Des
ktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-
packages\sklearn\metrics\_scorer.py", line 53, in _cached_call
    return getattr(estimator, method)(*args, **kwargs)
AttributeError: 'MajorityVoteClassifier' object has no attribute 'predict_proba'

  UserWarning,
```

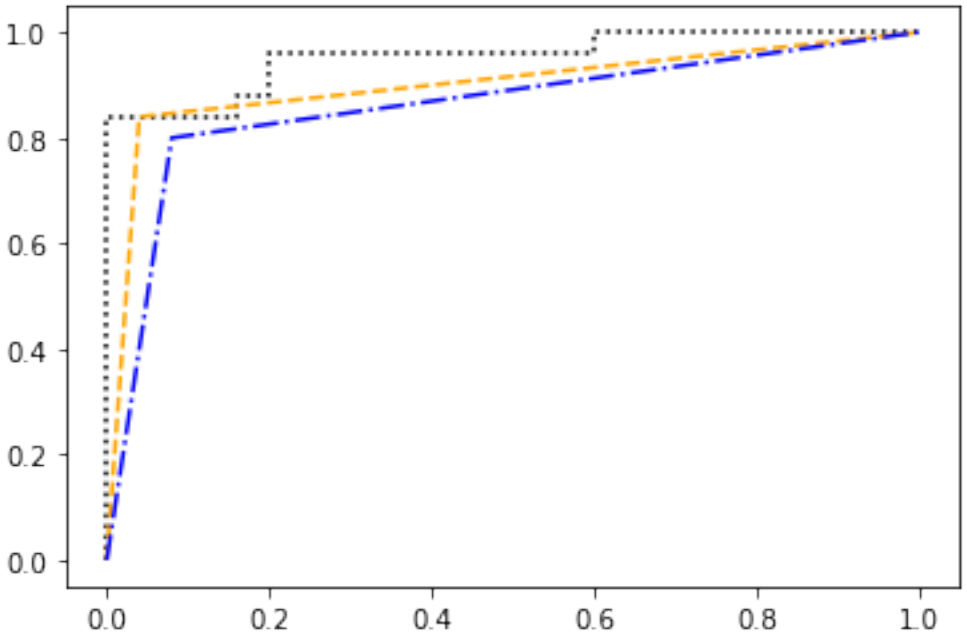
0.4 Evaluating and tuning Ensemble Classifier

[13]:
```
from sklearn.metrics import roc_curve
from sklearn.metrics import auc
colors = ['black', 'orange', 'blue', 'green']
linestyles = [':', '--', '-.', '-']
for clf, label, clr, ls in zip(all_clf, clf_labels, colors, linestyles):
    # assuming the label of the positive class is 1
    y_pred = clf.fit(X_train,y_train).predict_proba(X_test)[:, 1]
    fpr, tpr, thresholds = roc_curve(y_true=y_test,y_score=y_pred)
    roc_auc = auc(x=fpr, y=tpr)
    plt.plot(fpr, tpr,color=clr,linestyle=ls,label='%s (auc = %0.2f)' % (label,␣
 ↪roc_auc))
plt.legend(loc='lower right')
plt.plot([0, 1], [0, 1],linestyle='--',color='gray',linewidth=2)
plt.xlim([-0.1, 1.1])
plt.ylim([-0.1, 1.1])
plt.grid(alpha=0.5)
plt.xlabel('False positive rate (FPR)')
plt.ylabel('True positive rate (TPR)')
plt.show()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-13-e7c30894c741> in <module>
      5 for clf, label, clr, ls in zip(all_clf, clf_labels, colors, linestyles)
      6     # assuming the label of the positive class is 1
----> 7     y_pred = clf.fit(X_train,y_train).predict_proba(X_test)[:, 1]
      8     fpr, tpr, thresholds = roc_curve(y_true=y_test,y_score=y_pred)
```

```
      9     roc_auc = auc(x=fpr, y=tpr)

AttributeError: 'MajorityVoteClassifier' object has no attribute 'predict_proba
```

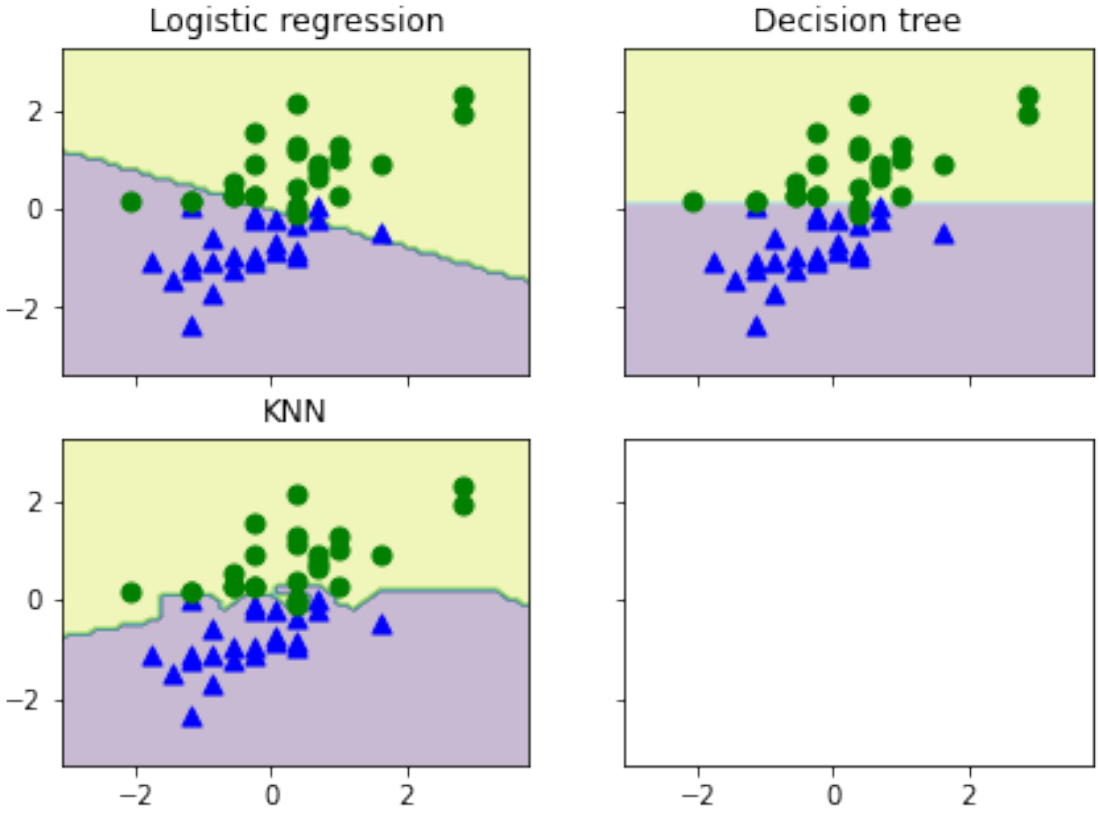


[14]:
```
sc = StandardScaler()
X_train_std = sc.fit_transform(X_train)
from itertools import product
x_min = X_train_std[:, 0].min() - 1
x_max = X_train_std[:, 0].max() + 1
y_min = X_train_std[:, 1].min() - 1
y_max = X_train_std[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.1),np.arange(y_min, y_max, 0.1))
f, axarr = plt.subplots(nrows=2, ncols=2,sharex='col',sharey='row',figsize=(7,␣
 ↪5))
for idx, clf, tt in zip(product([0, 1], [0, 1]),all_clf, clf_labels):
    clf.fit(X_train_std, y_train)
    Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
    axarr[idx[0], idx[1]].contourf(xx, yy, Z, alpha=0.3)
    axarr[idx[0], idx[1]].scatter(X_train_std[y_train==0,␣
 ↪0],X_train_std[y_train==0, 1],c='blue',marker='^',s=50)
    axarr[idx[0], idx[1]].scatter(X_train_std[y_train==1,␣
 ↪0],X_train_std[y_train==1, 1],c='green',marker='o',s=50)
    axarr[idx[0], idx[1]].set_title(tt)
```

```
plt.text(-3.5, -4.5,s='Sepal width [standardized]',ha='center', va='center',␣
 ↪fontsize=12)
plt.text(-10.5, 4.5,s='Petal length [standardized]',ha='center',␣
 ↪va='center',fontsize=12, rotation=90)
plt.show()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-14-711bdbdbff4d> in <module>
     10 for idx, clf, tt in zip(product([0, 1], [0, 1]),all_clf, clf_labels):
     11     clf.fit(X_train_std, y_train)
---> 12     Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
     13     Z = Z.reshape(xx.shape)
     14     axarr[idx[0], idx[1]].contourf(xx, yy, Z, alpha=0.3)

AttributeError: 'MajorityVoteClassifier' object has no attribute 'predict'
```



[ ]:
```
mv_clf.get_params()
```

[15]:
```
from sklearn.model_selection import GridSearchCV
params = {'decisiontreeclassifier__max_depth': [1, 2],'pipeline-1__clf__C': [0.
 ↪001, 0.1, 100.0]}
grid = GridSearchCV(estimator=mv_clf,param_grid=params,cv=10,scoring='roc_auc')
grid.fit(X_train, y_train)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-15-6ec07cf54281> in <module>
      2 params = {'decisiontreeclassifier__max_depth': [1,␣
 ↪2],'pipeline-1__clf__C': [0.001, 0.1, 100.0]}
      3 grid =␣
 ↪GridSearchCV(estimator=mv_clf,param_grid=params,cv=10,scoring='roc_auc')
----> 4 grid.fit(X_train, y_train)

~\OneDrive - Reliance Corporate IT Park␣
 ↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
 ↪py in inner_f(*args, **kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64
     65             # extra_args > 0

~\OneDrive - Reliance Corporate IT Park␣
 ↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
 ↪py in fit(self, X, y, groups, **fit_params)
    839                 return results
    840
--> 841             self._run_search(evaluate_candidates)
    842
    843             # multimetric is determined here because in the case of a␣
 ↪callable

~\OneDrive - Reliance Corporate IT Park␣
 ↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
 ↪py in _run_search(self, evaluate_candidates)
   1294     def _run_search(self, evaluate_candidates):
   1295         """Search all candidates in param_grid"""
-> 1296         evaluate_candidates(ParameterGrid(self.param_grid))
   1297
   1298

~\OneDrive - Reliance Corporate IT Park␣
 ↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
 ↪py in evaluate_candidates(candidate_params, cv, more_results)
    807                                 (split_idx, (train, test)) in product(
    808                                 enumerate(candidate_params),
```

```
--> 809                                     enumerate(cv.split(X, y, groups))))
    810
    811                 if len(out) < 1:

~\OneDrive - Reliance Corporate IT Park␣
 ↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
 ↪py in __call__(self, iterable)
   1046             # remaining jobs.
   1047             self._iterating = False
-> 1048             if self.dispatch_one_batch(iterator):
   1049                 self._iterating = self._original_iterator is not None
   1050

~\OneDrive - Reliance Corporate IT Park␣
 ↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
 ↪py in dispatch_one_batch(self, iterator)
    862                 return False
    863             else:
--> 864                 self._dispatch(tasks)
    865                 return True
    866

~\OneDrive - Reliance Corporate IT Park␣
 ↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
 ↪py in _dispatch(self, batch)
    780         with self._lock:
    781             job_idx = len(self._jobs)
--> 782             job = self._backend.apply_async(batch, callback=cb)
    783             # A job can complete so quickly than its callback is
    784             # called before we get here, causing self._jobs to

~\OneDrive - Reliance Corporate IT Park␣
 ↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
 ↪py in apply_async(self, func, callback)
    206     def apply_async(self, func, callback=None):
    207         """Schedule a func to be run"""
--> 208         result = ImmediateResult(func)
    209         if callback:
    210             callback(result)

~\OneDrive - Reliance Corporate IT Park␣
 ↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
 ↪py in __init__(self, batch)
    570         # Don't delay the application, to avoid keeping the input
    571         # arguments in memory
--> 572         self.results = batch()
    573
    574     def get(self):
```

```
~\OneDrive - Reliance Corporate IT Park␣
↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
↪py in __call__(self)
    262         with parallel_backend(self._backend, n_jobs=self._n_jobs):
    263             return [func(*args, **kwargs)
--> 264                     for func, args, kwargs in self.items]

    265
    266     def __reduce__(self):

~\OneDrive - Reliance Corporate IT Park␣
↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
↪py in <listcomp>(.0)
    262         with parallel_backend(self._backend, n_jobs=self._n_jobs):
    263             return [func(*args, **kwargs)
--> 264                     for func, args, kwargs in self.items]

    265
    266     def __reduce__(self):

~\OneDrive - Reliance Corporate IT Park␣
↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
↪py in __call__(self, *args, **kwargs)
    220     def __call__(self, *args, **kwargs):
    221         with config_context(**self.config):
--> 222             return self.function(*args, **kwargs)

~\OneDrive - Reliance Corporate IT Park␣
↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
↪py in _fit_and_score(estimator, X, y, scorer, train, test, verbose,␣
↪parameters, fit_params, return_train_score, return_parameters,␣
↪return_n_test_samples, return_times, return_estimator, split_progress,␣
↪candidate_progress, error_score)
    584             cloned_parameters[k] = clone(v, safe=False)
    585
--> 586         estimator = estimator.set_params(**cloned_parameters)
    587
    588     start_time = time.time()

~\OneDrive - Reliance Corporate IT Park␣
↪Limited\Desktop\Self_Projects\Python_Machine_Learning_Sebastian_Raschka\myenv\lib\site-pack
↪py in set_params(self, **params)
    231                                 'Check the list of available parameter␣
↪'
    232                                 'with `estimator.get_params().keys()`.␣
↪%
--> 233                                 (key, self))
    234
    235             if delim:
```

```
ValueError: Invalid parameter decisiontreeclassifier for estimator␣
↪MajorityVoteClassifier(classifiers=[Pipeline(steps=[('sc', StandardScaler()),
                                                     ['clf',
                                                      LogisticRegression(C=0.001
                                                                         ␣
↪random_state=1)]]),
                                      DecisionTreeClassifier(criterion='entropy',
                                                             max_depth=1,
                                                             random_state=0),
                                      Pipeline(steps=[('sc', StandardScaler()),
                                                      ['clf',
                                                       ␣
↪KNeighborsClassifier(n_neighbors=1)]])]). Check the list of available␣
↪parameters with `estimator.get_params().keys()`.
```

[16]:
```
for params, mean_score, scores in grid.grid_scores_:
    print("%0.3f+/-%0.2f %r"% (mean_score, scores.std() / 2, params))
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-16-efe2877d08b1> in <module>
----> 1 for params, mean_score, scores in grid.grid_scores_:
      2     print("%0.3f+/-%0.2f %r"% (mean_score, scores.std() / 2, params))

AttributeError: 'GridSearchCV' object has no attribute 'grid_scores_'
```

[17]:
```
print('Best parameters: %s' % grid.best_params_)
print('Accuracy: %.2f' % grid.best_score_)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-17-e3da298ba977> in <module>
----> 1 print('Best parameters: %s' % grid.best_params_)
      2 print('Accuracy: %.2f' % grid.best_score_)

AttributeError: 'GridSearchCV' object has no attribute 'best_params_'
```

0.5 Bagging : Building an ensemble of classifiers from bootstrap samples

0.5.1 Applying bagging to classify samples in the Wine dataset

Here, we will only consider the Wine classes 2 and 3, and we select two features: Alcohol and OD280/OD315 of diluted wines:

[18]:
```
import pandas as pd
df_wine = pd.read_csv('https://archive.ics.uci.edu/ml/
 ↪machine-learning-databases/wine/wine.data',header=None)
df_wine.columns = ['Class label', 'Alcohol','Malic acid', 'Ash','Alcalinity of␣
 ↪ash','Magnesium', 'Total phenols','Flavanoids', 'Nonflavanoid␣
 ↪phenols','Proanthocyanins','Color intensity', 'Hue','OD280/OD315 of diluted␣
 ↪wines','Proline']
# drop 1 class
df_wine = df_wine[df_wine['Class label'] != 1]
y = df_wine['Class label'].values
X = df_wine[['Alcohol','OD280/OD315 of diluted wines']].values
```

Next, we encode the class labels into binary format and split the dataset into 80 percent training and 20 percent test sets, respectively:

[19]:
```
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
le = LabelEncoder()
y = le.fit_transform(y)
X_train, X_test, y_train, y_test =train_test_split(X, y,test_size=0.
 ↪2,random_state=1,stratify=y)
```

[20]:
```
from sklearn.ensemble import BaggingClassifier
tree = DecisionTreeClassifier(criterion='entropy',random_state=1,max_depth=None)
bag = BaggingClassifier(base_estimator=tree,n_estimators=500,max_samples=1.
 ↪0,max_features=1.
 ↪0,bootstrap=True,bootstrap_features=False,n_jobs=1,random_state=1)
```

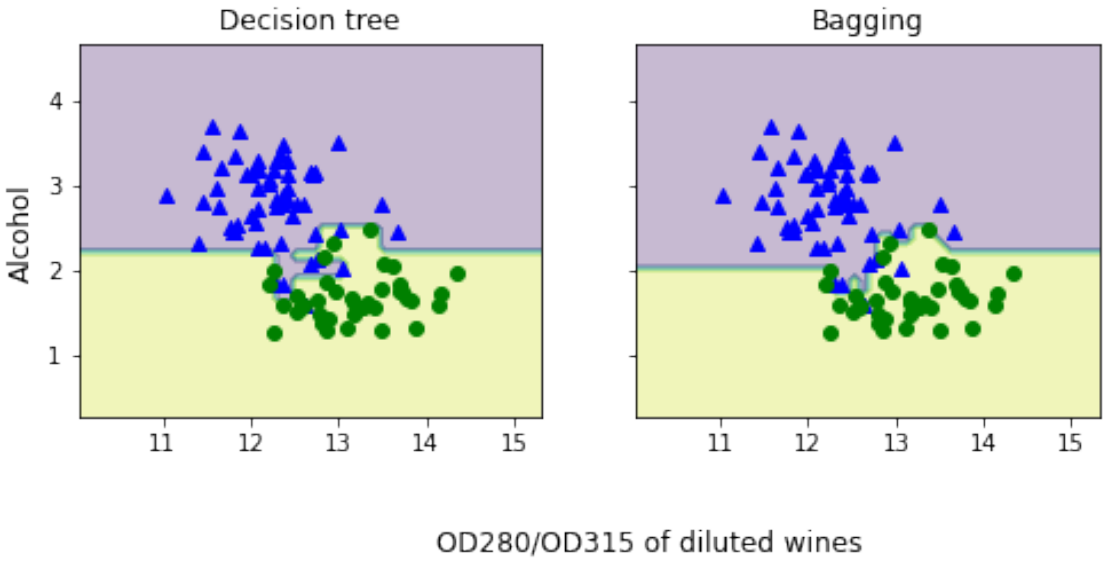
[21]:
```
from sklearn.metrics import accuracy_score
tree = tree.fit(X_train, y_train)
y_train_pred = tree.predict(X_train)
y_test_pred = tree.predict(X_test)
tree_train = accuracy_score(y_train, y_train_pred)
tree_test = accuracy_score(y_test, y_test_pred)
print('Decision tree train/test accuracies %.3f/%.3f' % (tree_train, tree_test))
```

```
Decision tree train/test accuracies 1.000/0.833
```

[22]:
```
bag = bag.fit(X_train, y_train)
y_train_pred = bag.predict(X_train)
y_test_pred = bag.predict(X_test)
bag_train = accuracy_score(y_train, y_train_pred)
bag_test = accuracy_score(y_test, y_test_pred)
print('Bagging train/test accuracies %.3f/%.3f'% (bag_train, bag_test))
```

```
Bagging train/test accuracies 1.000/0.917
```

[24]:
```
x_min = X_train[:, 0].min() - 1
x_max = X_train[:, 0].max() + 1
y_min = X_train[:, 1].min() - 1
y_max = X_train[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.1),np.arange(y_min, y_max, 0.1))
f, axarr = plt.subplots(nrows=1, ncols=2,sharex='col',sharey='row',figsize=(8,␣
 ↪3))
for idx, clf, tt in zip([0, 1],[tree, bag],['Decision tree', 'Bagging']):
    clf.fit(X_train, y_train)
    Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
    axarr[idx].contourf(xx, yy, Z, alpha=0.3)
    axarr[idx].scatter(X_train[y_train==0, 0],X_train[y_train==0, 1],c='blue',␣
 ↪marker='^')
    axarr[idx].scatter(X_train[y_train==1, 0],X_train[y_train==1, 1],c='green',␣
 ↪marker='o')
    axarr[idx].set_title(tt)
axarr[0].set_ylabel('Alcohol', fontsize=12)
plt.text(10.2, -1.2,s='OD280/OD315 of diluted wines',ha='center', va='center',␣
 ↪fontsize=12)
plt.show()
```



0.6 Leveraging weak learners via adaptive boosting

0.6.1 Applying AdaBoost using scikit-learn

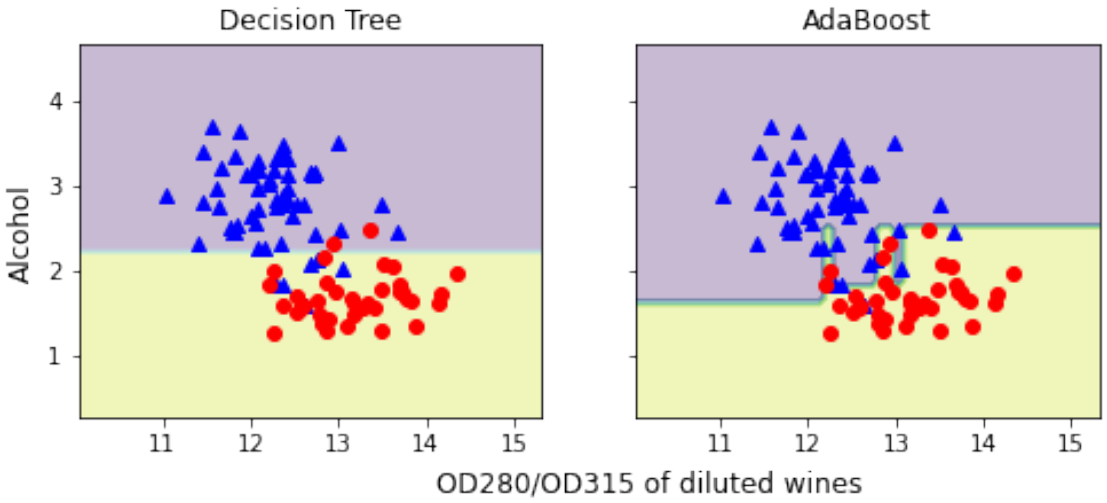
[25]:
```
from sklearn.ensemble import AdaBoostClassifier
tree = DecisionTreeClassifier(criterion='entropy',random_state=1,max_depth=1)
ada = AdaBoostClassifier(base_estimator=tree,n_estimators=500,learning_rate=0.
 ↪1,random_state=1)
tree = tree.fit(X_train, y_train)
y_train_pred = tree.predict(X_train)
y_test_pred = tree.predict(X_test)
tree_train = accuracy_score(y_train, y_train_pred)
tree_test = accuracy_score(y_test, y_test_pred)
print('Decision tree train/test accuracies %.3f/%.3f'% (tree_train, tree_test))
```

```
Decision tree train/test accuracies 0.916/0.875
```

[26]:
```
ada = ada.fit(X_train, y_train)
y_train_pred = ada.predict(X_train)
y_test_pred = ada.predict(X_test)
ada_train = accuracy_score(y_train, y_train_pred)
ada_test = accuracy_score(y_test, y_test_pred)
print('AdaBoost train/test accuracies %.3f/%.3f'% (ada_train, ada_test))
```

```
AdaBoost train/test accuracies 1.000/0.917
```

[27]:
```
x_min = X_train[:, 0].min() - 1
x_max = X_train[:, 0].max() + 1
y_min = X_train[:, 1].min() - 1
y_max = X_train[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.1),np.arange(y_min, y_max, 0.1))
f, axarr = plt.subplots(1, 2,sharex='col',sharey='row',figsize=(8, 3))
for idx, clf, tt in zip([0, 1],[tree, ada],['Decision Tree', 'AdaBoost']):
    clf.fit(X_train, y_train)
    Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
    axarr[idx].contourf(xx, yy, Z, alpha=0.3)
    axarr[idx].scatter(X_train[y_train==0, 0],X_train[y_train==0,␣
 ↪1],c='blue',marker='^')
    axarr[idx].scatter(X_train[y_train==1, 0],X_train[y_train==1,␣
 ↪1],c='red',marker='o')
    axarr[idx].set_title(tt)
    axarr[0].set_ylabel('Alcohol', fontsize=12)
plt.text(10.2, -0.5,s='OD280/OD315 of diluted␣
 ↪wines',ha='center',va='center',fontsize=12)
plt.show()
```


Decision Tree
AdaBoost
Alcohol
4
3
2
1
11
12
13
14
15
OD280/OD315 of diluted wines

[ ]: