Link to GitHub: https://github.com/TakumiDawn/Machine-Learning-in-Finance-Lab

In [1]:
```
#using KNN and DT for the iris dataset
import numpy as np
from sklearn import datasets, preprocessing
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
```

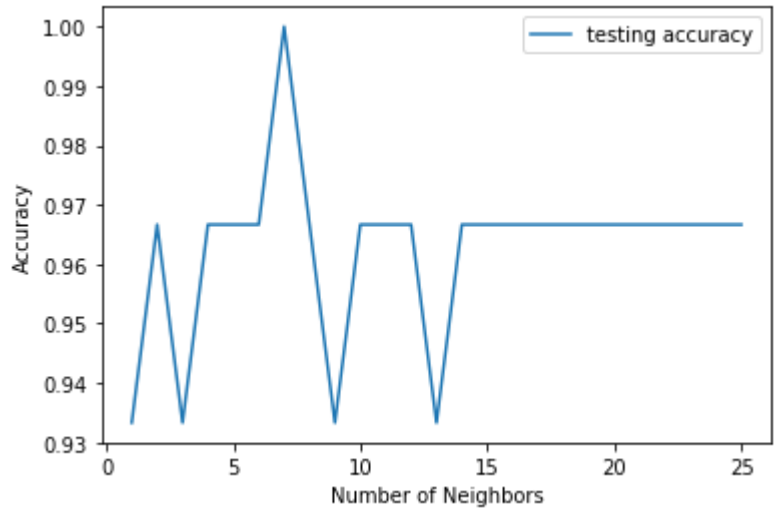
In [2]:
```
#split the dataset
iris = datasets.load_iris()
X = iris.data[:, [2, 3]]
y = iris.target

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=33,stra
```

In [3]:
```
# Standardizing the features:
scaler=StandardScaler().fit(X_train)
X_train=scaler.transform(X_train)
X_test=scaler.transform(X_test)
X_combined_std = np.vstack((X_train, X_test))
y_combined = np.hstack((y_train, y_test))
```

In [4]:
```
from sklearn.neighbors import KNeighborsClassifier
```

In [5]:
```
#KNN Training
k_range=range(1,26)
score=[]
for k in k_range:
    knn=KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)
    y_pred=knn.predict(X_test)
    y_train_pred=knn.predict(X_train)
    score.append(accuracy_score(y_test,y_pred))
```

In [6]:
```
import matplotlib.pyplot as plt
plt.plot(range(1,26),score, label='testing accuracy')
plt.legend()
plt.xlabel('Number of Neighbors')
plt.ylabel('Accuracy')
plt.show()
```



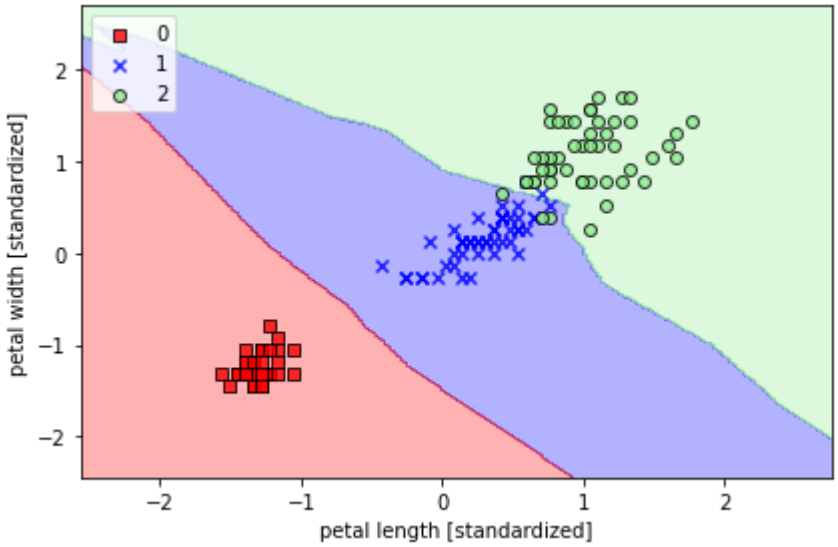
In [7]:
```
print('accuracy:',max(score),'; best K:', score.index(max(score)))
```

```
accuracy: 1.0 ; best K: 6
```

In [8]:

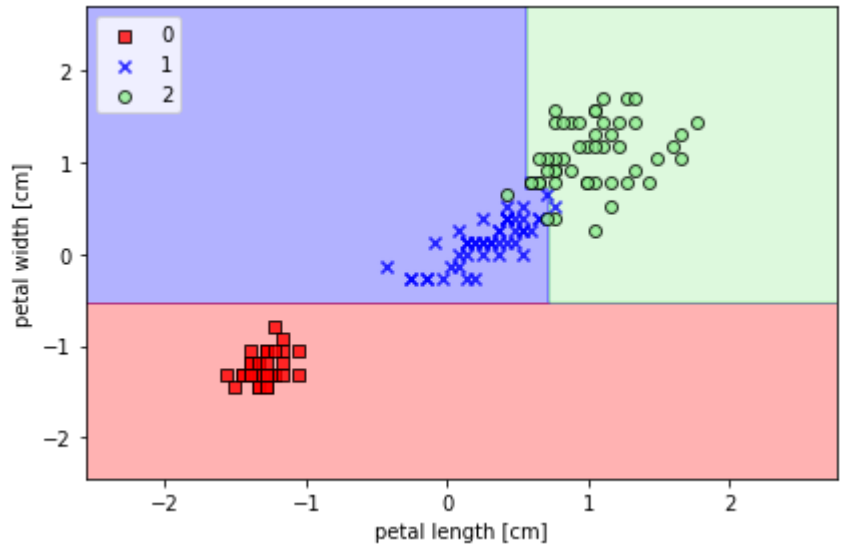
```
from matplotlib.colors import ListedColormap
def plot_decision_regions(X, y, classifier, test_idx=None, resolution=0.02):

    # setup marker generator and color map
    markers = ('s', 'x', 'o', '^', 'v')
    colors = ('red', 'blue', 'lightgreen', 'gray', 'cyan')
    cmap = ListedColormap(colors[:len(np.unique(y))])

    # plot the decision surface
    x1_min, x1_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    x2_min, x2_max = X[:, 1].min() - 1, X[:, 1].max() + 1
    xx1, xx2 = np.meshgrid(np.arange(x1_min, x1_max, resolution),
                           np.arange(x2_min, x2_max, resolution))
    Z = classifier.predict(np.array([xx1.ravel(), xx2.ravel()]).T)
    Z = Z.reshape(xx1.shape)
    plt.contourf(xx1, xx2, Z, alpha=0.3, cmap=cmap)
    plt.xlim(xx1.min(), xx1.max())
    plt.ylim(xx2.min(), xx2.max())

    for idx, cl in enumerate(np.unique(y)):
        plt.scatter(x=X[y == cl, 0],
                    y=X[y == cl, 1],
                    alpha=0.8,
                    c=colors[idx],
                    marker=markers[idx],
                    label=cl,
                    edgecolor='black')
```

In [9]:
```
# KNN
knn = KNeighborsClassifier(n_neighbors=6,# best k = 6
                           p=2,
                           metric='minkowski')
knn.fit(X_train, y_train)

plot_decision_regions(X_combined_std, y_combined,
                      classifier=knn, test_idx=range(105, 150))

plt.xlabel('petal length [standardized]')
plt.ylabel('petal width [standardized]')
plt.legend(loc='upper left')
plt.tight_layout()
plt.show()
```



In [10]:
```
## Decision tree
from sklearn.tree import DecisionTreeClassifier
from matplotlib.colors import ListedColormap
```

In [11]:
```
tree = DecisionTreeClassifier(criterion='gini',
                              max_depth=4,
                              random_state=1)
tree.fit(X_train, y_train)

X_combined = np.vstack((X_train, X_test))
y_combined = np.hstack((y_train, y_test))
plot_decision_regions(X_combined, y_combined,
                      classifier=tree, test_idx=range(105, 150))

plt.xlabel('petal length [cm]')
plt.ylabel('petal width [cm]')
plt.legend(loc='upper left')
plt.tight_layout()
plt.show()
```



In [12]:
```
print("My name is Takumi Li")
print("My NetID is: feiyang3")
print("I hereby certify that I have read the University policy on Academic Integrity a
```

```
My name is Takumi Li
My NetID is: feiyang3
I hereby certify that I have read the University policy on Academic Integrity and tha
t I am not in violation.
```